2017

Strategies for Combining Tree-Based Ensemble Models

Yi Zhang
Nova Southeastern University, yi_zhang408@hotmail.com

Strategies for Combining Tree-Based Ensemble Models

by

Yi Zhang

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in
Information Systems

College of Engineering and Computing
Nova Southeastern University

2017

We hereby certify that this dissertation, submitted by Yi Zhang, conforms to acceptable standards and is fully adequate in scope and quality to fulfill the dissertation requirements for the degree of Doctor of Philosophy.

Sumitra Mukherjee, Ph.D.
Chairperson of Dissertation Committee

November 28, 2017
Date

Michael J. Laszlo, Ph.D.
Dissertation Committee Member

November 28, 2017
Date

Francisco J. Mitropoulos, Ph.D.
Dissertation Committee Member

November 28, 2017
Date

Approved:

Yong X. Tao, Ph.D., P.E., FASME
Dean, College of Engineering and Computing

November 28, 2017
Date

College of Engineering and Computing
Nova Southeastern University

2017

An Abstract of a Dissertation Submitted to Nova Southeastern University
In Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

Strategies for Combining Tree-Based Ensemble Models

by
Yi Zhang
October 2017

Ensemble models have proved effective in a variety of classification tasks. These models combine the predictions of several base models to achieve higher out-of-sample classification accuracy than the base models. Base models are typically trained using different subsets of training examples and input features. Ensemble classifiers are particularly effective when their constituent base models are diverse in terms of their prediction accuracy in different regions of the feature space. This dissertation investigated methods for combining ensemble models, treating them as base models. The goal is to develop a strategy for combining ensemble classifiers that results in higher classification accuracy than the constituent ensemble models. Three of the best performing tree-based ensemble methods – random forest, extremely randomized tree, and eXtreme gradient boosting model – were used to generate a set of base models. Outputs from classifiers generated by these methods were then combined to create an ensemble classifier. This dissertation systematically investigated methods for (1) selecting a set of diverse base models, and (2) combining the selected base models. The methods were evaluated using public domain data sets which have been extensively used for benchmarking classification models. The research established that applying random forest as the final ensemble method to integrate selected base models and factor scores of multiple correspondence analysis turned out to be the best ensemble approach.

Acknowledgment

I would like to thank my advisor, Professor Sumitra Mukherjee, for his excellent guidance and support of the research. I also thank the committee members, Professor Michael J. Laszlo and Professor Francisco J. Mitropoulos, without whose support I would not have been able to finish the dissertation.

To my beloved family members, Peixiang Liu, Hugo Liu, and Rickey Liu: I would like to thank you for your truly support and love. It was always helpful to share my happiness and relaxation during breaks of my research. A particular note of thanks goes to my parents. Their wise counsel and kind words motivated me as always.

Table of Contents

List of Tables

Tables

List of Figures

Figures

Chapter 1

Introduction

Background

An ensemble approach integrates the output of a group of machine learning algorithms. The purpose of an ensemble approach is to achieve an improved classification accuracy that outperforms the individual learning algorithms which are often called base models. It has been shown that ensemble-based learning algorithms improve the predictive accuracy in many applications (Banfield, Hall, Bowyer, & Kegelmeyer, 2007; Leblanc & Tibshirani, 1996; Rodrigues, Kuncheva, & Alonso, 2006). Combining multiple learning algorithms has been found to be effective for various problems (Breiman, 2001; Zhang et al., 2011; Zhu, Beling, & Overstreet, 2002).

The initial step in an ensemble approach is creating various base models (Dietterich, 2001). The individual base models should be diverse enough in the sense that they have minimum errors in common. Base models can be generated 1) by different learning methods, 2) by using sub-samples of training data set, or/and 3) by using subsets of attributes or input features. Base models are generated by applying those three methods individually or together. Researchers in statistics and machine learning focus on constructing ensembles in which multiple base classifiers are

generated by perturbing or splitting the training data set. The training subsets are random samples with replacement or without replacement from the original training data. Several well-known ensemble-based learning algorithms, such as bagging, boosting, and random forest, have been widely accepted and applied for prediction tasks (Breiman, 1996 & 2001; Freund & Shapire, 1996). They have been shown to have consistently better performance than non-ensemble-based models.

The random forest (RF), extremely randomized trees (ERT), and extreme gradient boosting (XGB) models were applied in this dissertation to generate base models due to their high predictive accuracy (Brieman, 2001; Caruana & Niculescu, 2006; Geurts, Ernst, & Wehenkel, 2006; Friedman, 2001). They are all tree-based and ensemble-based machine learning algorithms. The RF model creates a large number of trees as base models by randomly selecting a subset of attributes in each splitting on randomly selected subsets of the training data (Brieman, 2001). Extremely randomized trees is a model similar to random forest. However, extremely randomized trees builds base classifiers on the whole training data by applying random selection on not only attributes but also the cut-point choice when splitting a tree node (Geurts, Ernst, & Wehenkel, 2006). The gradient boosting algorithm is an ensemble method in which the final classifier is combined by weak classifiers step by step (Friedman, 2001). In gradient boosting, a differentiable loss function is used to calculate the adjustments to the consecutive success learner in an iterative learning sequence. It assigns higher weights to misclassified observations when creating the subsequent tree. XGB is a scalable implementation of gradient boosting which is a very time efficient algorithm (Friedman, 2001; Friedman, Hastie, & Tibshirani 2000). By considering both training

loss and regularization, XGB can quickly reach the optimal decision and control overfitting at the same time.

Most commonly, all base models are ensembled together for the final output. However, researchers showed that combining a subset of base models with desirable characteristics worked better than combining all models (Ruta & Gabrys, 2005; Zhu, 2010). Selecting only a subset of base models might also contribute to both the accuracy of the final decision and the computing efficiency (Tsoumakas, Partalas, & Vlahavas, 2008). Jurek, Bi, Wu, and Nugent (2013) categorized base model selection techniques into static selection and dynamic selection. In static selection, the same subset of base models is used for both training and testing data sets (Zhu, 2010). While in dynamic selection, a subgroup of base models that locally perform better are chosen to make the decision (Cevikalp & Polikar, 2008). Base models can be selected based on either accuracy or diversity or both of these criteria (Jurek, Bi, Wu, & Nugent, 2003; Hu, 2001). Since the ensemble-based models, RF, ERT, and XGBoost as base models usually achieve good classification accuracy, this research focuses on applying correlation analysis and backward selection on the output of base models to identify an optimal subset of diverse base models, and multiple correspondence analysis (MCA) to capture the features of outputs of base models, thus to achieve more accurate predictions (Abdelazeem, 2008; Ruta and Gabrys, 2005).

After base models are selected, how to combine base models is the question to be addressed next. Researchers must consider and decide the kind of information to be integrated and the combining method to be applied. Generally, an ensemble approach integrates all or selected outputs of base models. The format of outputs from base

models varies, which can either be class label or probability. The combining technique can be majority voting, which is very effective when applying with a group of properly selected base models, such as decision trees in a random forest model (Breiman, 2001). It can also use various machine learning algorithms to integrate the outputs of base classifiers. For example, a logistic regression model is used to combine outputs of base models in stacking (Wolpert, 1992). Stacking, which is also called Stacked Generalization, has proven to be one of the most effective ensemble methods that improves the accuracy of the final decision of both classification and regression problems (Dzeroski & Zenko, 2004; Seewald, 2002; Jurek, Bi, Wu, & Nugent, 2001).

In this research, we chose random forest, extremely randomized trees, and extreme gradient boosting to construct base classifiers, applied model selection techniques, and integrated classifiers using various machine learning algorithms (random forest, logistic regression, and extreme gradient boosting). We systematically investigated the decision accuracy of the base models RF, ERT and XGB; how model selection techniques impacted the final ensemble result; the relationship between model combination techniques and the final ensemble results; and whether there existed a better ensemble approach.

Problem Statement

Improving predictive accuracy of machine learning algorithms is an ongoing research challenge. Numerous studies have shown that ensemble techniques increase the predictive accuracy when compared with non-ensemble-based classifiers for both classification and regression problem (Breiman, 1996; Dietterich, 2000; Leblanc and Tibshirani, 1996; Zhu, 2010). The majority of the related studies focused on integrating

weak classifiers, such as decision trees that were generated by perturbing the training data set (Breiman, 2000; Zhu, 2010). Researchers also demonstrated that picking several best models worked better than combining all models under some circumstances (Kotsiantis, 2011; Russell & Adam, 1987). The best models can be either those with various local predictive powers or those with the best predictive accuracy. Ensemble classifiers are particularly effective when the constituent base models are diverse in terms of their prediction accuracy in different regions of the feature space. The investigation of how to combine these ensemble-based models is a major research topic in the field of machine learning (Kotsiantis, 2011). In this dissertation, we studied methods for combining ensemble models by treating them as base models. Three tree-based ensemble methods – random forest, extremely randomized trees, and extreme gradient boosting model – were used to generate a set of base models (Brieman, 2001; Geurts, Ernst, & Wehenkel, 2006; Friedman, 2001). Outputs from classifiers generated by these methods were then combined to create an ensemble classifier to provide the final prediction. We systematically investigated methods for (1) selecting a set of diverse base models, and (2) combining the selected base models. The selection and combination methods were evaluated using public domain data sets which have been extensively used for benchmarking classification models.

Dissertation Goal

The goal of this dissertation is to develop a strategy for combining ensemble classifiers that results in higher classification accuracy than the constituent ensemble models. We investigated ensemble approaches which used random forest, extremely randomized trees, and extreme gradient boosting algorithm to generate base models.

Performances of base models were evaluated and compared. Correlation of outputs of base models were examined using Cramer's V correlation analysis. Various base model selection techniques based on correlation or accuracy of base models were applied and compared. Different model combination techniques, majority voting if applicable, logistic regression, extreme gradient boosting, and random forest, were applied to all or optimal subsets of base classifiers. The performance of final ensemble outputs was evaluated.

Research Questions

1. Will specific ensemble approaches of ensemble-based models increase the predictive accuracy compared with extant single ensemble models?
2. Are random forest, extremely randomized trees, and extreme gradient boosting good candidates as base classifiers?
3. Will various model selection techniques make a difference in the predictive accuracy of the overall ensemble approach?
4. How will various model combination techniques affect the predictive accuracy of the ensemble approach?

Chapter 2

Review of the Literature

Overview

An ensemble approach starts from creating various base classifiers, selecting base classifiers, and ends in combining base classifiers. Various investigations have demonstrated that ensemble approaches of different classifiers improve the accuracy of the final classifier (Parvin & Alizadeh, 2011). Researchers evaluate learning algorithms by investigating the variance and bias (Kohavi & Wolpert, 1996). Variance measures the difference of prediction of a learning algorithm on different data sets. Bias measures the average error of a classifier trained with different training data sets. A single classifier usually has large bias and little variance when compared with a group of integrated classifiers (Webb & Conilione, 2003). It has been demonstrated that ensemble approaches usually reduce either variance or bias or both (Bauer & Kohavi, 1999).

The decision tree learning algorithm is a flowchart-like model that is widely used by researchers in information systems and machine learning. A decision tree model usually shows high variance in both choosing attributes and splitting nodes (Breiman, Friedman, Olshen, & Stone, 1984). It has been experimentally shown that cut-point variance of a decision tree model is extremely high for both small and large

data sets (Wehenkel, 1997; Geurts, 2000). The cut-point variance rephrases part of the error rate of the learning process. Because of the high variance, the decision tree is considered an unstable classifier. However, it works very well as the base classifier in ensemble approaches (Brieman, 2002). Several well-known ensemble approaches, such as boosting, random forest, and extremely randomized trees which incorporate a decision tree algorithm as the base models, are very successful in generating higher predictive accuracy (Breiman, 2002; Freund & Shapire, 1996). The idea behind these ensemble approaches is to reduce the variance of the learning algorithm without increasing the bias too much. These ensemble algorithms bring randomization into generating the same type of base classifiers (decision trees) on randomized training data sets. They generally are very competitive in producing better predictive accuracy than other non-ensemble-based machine learning algorithms (Dietterich, 2000).

It has been demonstrated that an ensemble model might avoid the mistake of choosing a wrong single model by statistically combining the output of base models, avoiding getting stuck in local optima computationally, and increasing the searching space for the true hypothesis (Dietterich, 2000). To avoid getting stuck in local optima and to increase the search space, diverse learning algorithms were often considered by researchers to include in the pool of base models (Kuncheva & Whitaker, 2003). Studies have demonstrated that the diversity of learning algorithms improves the accuracy of an ensemble approach (Dietterich, 2000). Diversity can be measured in various ways. A major measurement is to test the correlation of the decision output of each base model. The group of less correlated models tends to provide higher predictive accuracy (Hu, 2001). A different technique to evaluate the diversity of base models is

Q statistics test (Kuncheva & Whitaker, 2003). Clustering the outputs of base models and then adding clusters as additional attributes to the training dataset, and random selection of attributes or instances are proven techniques to increase the diversity of base models (Bryll, Gutierrez-Osuna, & Quek, 2003; Gan & Xiao, 2009). Among the stated methods, creating base models by randomly selecting either attributes or instances or both has been widely applied and has achieved tremendous success (Brieman, 2001).

Ensemble Models

Bagging

Brieman first proposed the idea of bagging which trained diverse individual base models by randomly selecting instances with replacement as training subsets (Breiman, 1996). It incorporates the idea of random selection which works by randomly selecting subsets of the training data set, manipulating the distribution of training data, or randomly selecting attributes (Breiman, 1996 & 2001; Freund & Shapire, 1996). Bagging is designed to reduce the variance of misclassification probability. Since base models can be trained independently, bagging can be very time-efficient. However, because of its strategy to create training data sets, bagging tends to improve the predictive accuracy by utilizing unstable classifiers, such as decision trees or artificial neural networks (Dietterich, 2000; Maclin, 1997). It has been shown that bagging is not able to improve the performance when using stable base models, such as linear regression (Skurichina, & Duin, 1998). Breiman (1996) explained that unstable models could be very diverse because they were sensitive to small changes of training data.

The diversity of base models is the key advantage of the bagging method to increase the final performance. However, at the same time, diversity also implies the unstable prediction of the randomly created base models. In order to obtain the same accuracy as an original decision tree, Machova and Barcak (2006) reported that the minimum number of base models in bagging should be twenty. Studies also reveal that bagging works more efficiently for small data sets (Skurichina, Kuncheva, & Duin, 2002).

Random Forest

Brieman (2001) proposed another ensemble approach, Random Forest, based on the idea of expanding diversity of base models by partitioning the attribute space. Random forest usually pools a lot of decision trees as base classifiers. It creates random training data sets for each individual decision tree by bootstrapping from the original training data set. It chooses the optimal attributes from a randomly selected subset of attributes at each split when growing a decision tree.

Random forest is an expanded version of bagging. The random subsets of instances don't have the same number of instances as the original training set. Generally, each subset has two thirds of the instances of the whole training data. At each split, an optimal attribute is chosen from around two thirds of the randomly selected attributes. Random forest not only adopts the advantages of bagging, such as more diversity of base classifiers and computational efficiency, but also overcomes some weaknesses of bagging, such as dealing with both small and large data very efficiently. Additionally, it is also designed to deal with the overfitting issue. Random forest is a very competitive and successful ensemble model and has been applied to

different research fields (Chi, Yeh, & Lai, 2011; Diaz-Uriarte & Alvarez de Andres, 2006).

Extremely Randomized Trees

Extremely Randomized Trees is another ensemble-based model. It is also called Extra Trees. It takes randomization even further when compared with random forest (Geurts et al., 2006). It randomizes not only the selection of instances and attributes, but also the selection of the cut point of splitting when growing individual base trees. The structures of total random trees are independent of the output of learning data. It is also extremely computationally efficient due to the extreme randomization.

The extremely randomized trees model works by decreasing variance while increasing bias at the same time. However, referenced to the standard decision tree model, if the randomization degree is optimal leveled, the variance can be extremely diminished and the bias increases only a little bit. The extremely randomized trees model has been demonstrated to be the top choice in many applications, such as high dimensional problems, mass-spectrometry datasets, and time series classification problems (Geurts & Wehenkel, 2005; Geurts, Fillet, De Seny, Meuwis, Mervilles, & Wehenkel, 2005b; Maree, Geurts, Piater, & Wehenkel, 2004). It has a very strong competitive predictive power, especially for classification problem, when compared with random forest and other ensemble approaches (Geurts et al., 2006).

Boosting

Another well-known ensemble technique, Boosting, was proposed by Freund and Schapire in 1996. It utilizes the random selection idea and manipulates the

distribution of training data by creating subsequent base models based on the predictive accuracy of previous base models. It is an iterative procedure that adds base classifiers one by one. Weights of one base model are calculated based on its predictive accuracy and are then applied when integrated with other base models. Weights of instance are also calculated by the current base model, and then are used to train the next base model. In this way, base models are regulated and the weighted predictions of the base models are combined to make the final decision. Boosting has been shown to reduce variance and bias (Rodriguez & Maudes, 2008). Good candidates for base models are decision trees or neural networks (Rodriguez & Maudes, 2008; Schwenk & Bengio, 2000).

Ada boosting is the benchmark model in boosting (Schapire, 1999). A number of studies have been explored to expand the techniques of Ada boosting to improve the accuracy and efficiency (Schapire, Freund, Bartlett, & LeeWS, 1998). Gradient boosting is one of its expansion forms and has earned a good reputation for its excellent performance in both accuracy and efficiency when compared with Ada boosting (Friedman, 2001). It utilizes a loss function to manipulate the adjustment that is applied to the subsequent base model. The training loss function measures how the model fits on training data. The gradient boosting model not only measures the model fit but also regulates the model complexity using a regularization function. Optimizing loss function tends to cause over-fitting. On the contrary, optimizing regularization function produces smaller variance for prediction. Balancing loss and regularization functions properly can produce optimal predictive performance and control the over-fitting issue (Johnson & Zhang, 2014).

Extreme gradient boosting is an algorithm created under the framework of gradient boosting (Chen, 2015). It utilizes generalized linear model and gradient boosted decision trees. Randomly sub-setting the instances and attributes techniques are applied in the extreme gradient boosting algorithm. It is very efficient in handling sparse matrices and producing accurate predictions.

Model Selection

Most ensemble approaches integrate all base models to make the final prediction. However, it has been shown that effective selection of a group of optimal base models based on diversity and accuracy can improve the final ensemble performance (Zeng, Chao, & Wong, 2010).

Abdelazeem (2008) proposed forward search or backward search methods to select an optimal set of base models based on majority voting error of the ensemble model. The forward search starts from the most accurate base model and adds other base models one by one until there is no improvement of predictive accuracy. The backward search starts from combining all of the base models, and then excludes base models one by one until the decrease of predictive accuracy is not acceptable.

Genetic algorithm (GA) has been applied in searching the best subset of base models when considering the accuracy of both base and final ensemble models (Kittler & Roli, 2001). Both diversity and accuracy are evaluated when applying the GA approach (Löfström, Johansson, & Bostrom, 2008). It is revealed that considering the accuracy of both the base and ensemble models is the most efficient approach for the GA approach of model selection.

Ruta and Gabrys (2005) selected the optimal subset of base models by evaluating various diversities of the models. In their approaches, diversities were presented by correlation coefficient, product moment correlation, Q statistics, disagreement measure, double-fault, entropy, and measure of difficulties. In addition to diversities, accuracies such as minimum individual error, mean error, and majority voting error were also considered. Various search methods, such as forward and backward search, random search, and GA search were explored. The experiment result showed that using majority voting error as the search criterion was the best way for model selection.

Model Integration

The last step in the ensemble approach is integrating the outputs of base models to make the final decision of regression or classification problems. The combination methods can be simple averaging, majority voting, or using functions or machine learning algorithms to combine base models (Brieman, 2001; Wolpert, 1992).

Majority voting is a simple but effective method, in which the final decision of an instance is voted by all base models. The case receiving the most votes is the final decision. An expanded version of majority voting is adding weights to base models where the weights are scaled by the accuracy or entropy of the base models. This weighting method has been expanded further by applying genetic algorithms (GA) to optimize the final result (Dimililer, Varoglu, & Altincay, 2007).

Stacked generalization is also an alternative way to combine multiple models (Wolpert, 1992). It works by reducing biases of learning algorithms with respect to a specific training data set. In stacked generalization, the outputs of base models for the

validating data set compose the training data for a meta-model. Then, a meta-learner is generated by a machine learning algorithm to combine the outputs of base models. Effective machine learning algorithm for the meta-learner can be multi-response linear regression and multi-response model tree (Dzeroski & Zenko, 2004; Seewald, 2002; Ting & Witten, 1999). Majority voting is not preferred in stacked generalization because it usually does not work on comparable or similar outputs of base models (Ting & Witten, 1999). However, the multilayer perceptron has been demonstrated to be an effective algorithm to combine the outputs of base models (Zhu, 2010). Logistic regression has also proved to be successful in combining outputs of base models in stacking (Wolpert, 1992).

Chapter 3

Methodology

Overview of Research Methodology

In this research, we explored an ensemble approach in which tree-based ensemble learning algorithms are used as base models. The primary goal is to study if an ensemble approach of ensemble-based models would further improve predictive accuracy. The secondary goal is investigating effective ways for selecting base models and various combination strategies. The overall ensemble procedure includes four major steps:

1. generating base models
2. calculating factor scores of multiple correspondence analysis
3. choosing optimal subsets of base models, and
4. integrating base models

Generating base models

Ten random forest, eleven extremely randomized trees (extra trees), and ten extreme gradient boosting models were generated to work as base models to ensemble. All base models are tree-based ensemble models (Brieman, 2001; Geurts et al., 2006; Freund & Schapire, 1996). They have proved to be relatively better models which

provide higher predictive accuracy. They all apply a randomization scheme to expand the diversity of base models in order to achieve better ensemble result.

Random Forests: The steps of building a random forest are listed as follows (Breiman, 2001).

1. To create a forest with K trees, K subsets of data are sampled with N instances randomly with replacement. Each subset grows one individual tree. Usually, the size of a subset is about two thirds of the size of the training set.
2. When building a single tree, at each splitting of node, m predictor variables are chosen randomly from all available variables. Each predictor is evaluated by a selected objective function. The one which provides the best splitting is used to do a binary split on that node. The same procedure is applied to all remaining nodes. The value of m can range from 1 to the total number of predictor variables. Most researchers set m to be the square root of the total number of predictor variables (Brieman, 2001).
3. K trees are created by repeating step 1 & 2 to construct a forest. When a new set of instances is input into the forest, one by one, each instance goes through every tree in the forest. The predictive result is the majority voting of the K trees for a classification problem.
4. Ten random forests were built on training data sets in this research. Seeds were randomly set up to ensure a repeatable predictive result for each forest. The number of individual trees in the forest ranged from 50 to 500. Because of the randomization strategy of sub-setting the instances and attributes and setting up different seeds and number of subtrees when building a random forest, these ten

forests had different structures and provided different predictions on the testing data set. Table 1 lists the number of trees of each base model.

Table 1

Number of Trees in Random Forest Base Model

Radom Forest	Number of Trees
1	50
2	100
3	150
4	200
5	250
6	300
7	350
8	400
9	450
10	500

Extremely Randomized Trees: The procedure of building extremely randomized trees, also called extra trees, is listed as follows (Geurts et al., 2006).

1. K decision trees are built without pruning from all training sample.
2. At each random splitting of a node, M predictor variables, $\{a_1, \dots, a_M\}$, among all non-constant candidate predictors are chosen without replacement and evaluated to split the node. M splits, $\{s_1, \dots, s_M\}$, one split per predictor, are generated from M predictors. A split s_* is selected if its score of evaluation is the most preferred one among all of the M splits. The same procedure is applied to each node.
3. Numerical predictors and categorical predictors follow different rules of splitting. For a categorical predictor a, A is used to denote its domain or the set of all possible values. A_S is a subset of A in which every value a appears in the training set S.

Then, a proper nonempty subset A_1 of A_S and a subset A_2 of $A \backslash A_S$ is randomly drawn. The split that meets $[a \in A_1 \cup A_2]$ is returned to compare with other splits. For a numerical predictor a, its maximal and minimal value in S, a_{min}^s and a_{max}^s, are calculated. A cutout point a_c is uniformly drawn in $[a_{min}^s, a_{max}^s]$. The split that meets $[a < a_c]$ is returned.

4. The K trees created by repeating step 2 & 3 are used to construct an extra trees model. When a new set of instances is input into the forest, one by one, each instance goes through all of the trees in the extra trees model. The result is the majority voting of the K trees for a classification problem.
5. Eleven extra trees models were built in this research. Seeds were randomly set up to ensure repeatable predictive result for each extra tree model. The number of individual trees in an extra trees model ranged from 50 to 550. Because of the randomization of sub-setting the instances and attributes, and setting up different seeds and number of subtrees when building an extra tree, these eleven extra trees had different structures and provided different predictions on the testing data set. Table 2 lists the number of trees in each model.

Table 2

Number of Trees in Extremely Randomized Trees Base Model

Extremely Randomized Trees	Number of Trees
1	50
2	100
3	150
4	200
5	250
6	300

Extremely Randomized Trees	Number of Trees
7	350
8	400
9	450
10	500
11	550

Extreme Gradient Boosting: Extreme gradient boosting model (XGB) is a tree-based ensemble model created under the gradient boosting framework proposed by Friedman (2001). Efficient linear solver and tree learning algorithm are implemented in XGB (Chen & He, 2015). The approach of constructing an XGB model is listed as follows (Chen & He, 2015).

1. XGB model is a summation of a collection of K weak trees. It is defined as $\sum_{k=1}^{K} f_k$, where f_k is the prediction of a decision tree.
2. Let x_i denote the feature vector for the i-th data point, the prediction with all the decision trees can be expressed as $\hat{y}_i = \sum_{k=1}^{K} f_k\,(x_i)$. In each iteration step, one tree is added to the collection, at the t-th step, the prediction is defined as $\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k\,(x_i)$.
3. When training the model, a loss function is chosen and optimized based on different types of task. For a binary classification problem, LogLoss is used as the loss function.

$$\mathrm{L} = -\frac{1}{N}\sum_{i=1}^{N}(y_i \log(p_i) + (1 - y_i)\log(1 - p_i))$$

where y_i is the real value of prediction on feature vector x_i, p_i is the probability on feature vector x_i, and N is the number of instances in the training data set. For a

multi-classification problem, mlogloss is used as the loss function which is defined as below, where C is the number of categories of target feature.

$$\mathrm{L} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{i,j}\log(p_{i,j})$$

4. When optimizing the loss function, XGB also implements a regularization term **Ω** to control the complexity in order to prevent overfitting.

$$\Omega = \mathrm{T} + \frac{1}{2}\lambda\sum_{j=1}^{T} w_j^2$$

T is the number of leaves. Instead of w_j, w_j^2 which is the score on the j-th leaf that is used for better controlling the complexity. λ and w both tune the complexity.

5. The objective function of XGB is defined as the combination of loss function and regularization. Loss function controls the predictive power and regularization controls the simplicity.

$$Obj = L + \Omega$$

6. Gradient descent is applied to optimize the objective function $Obj(y, \hat{y})$. It is an iterative technique that calculates $\partial_{\hat{y}} Obj(y, \hat{y})$ at each iteration. $\hat{y}$ is improved along the direction of the gradient to minimize the objective.

7. For an iterative algorithm, the objective function at each step can be rewritten as

$$Obj^{(t)} = \sum_{i=1}^{N} L(y_i, \hat{y}_i^{(t)}) + \sum_{i=1}^{t}\Omega(f_i) = \sum_{i=1}^{N} L(y_i, \hat{y}_i^{(t-1)}) + f_t(x_i) + \sum_{i=1}^{t}\Omega(f_i)$$

The first and second order gradient $\partial_{\hat{y}_i^{(t)}} Obj^{(t)}$ *and* $\partial^2_{\hat{y}_i^{(t)}} Obj^{(t)}$ are calculated to improve the performance. The Taylor approximation of the objective function is derived as follows since there might be no derivative for every objective function.

$$Obj^{(t)} \cong \sum_{i=1}^{N}[L(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \sum_{i=1}^{t} \Omega(f_i)$$

Where $g_i = \partial_{\hat{y}_i^{(t-1)}} \mathrm{L}\left(y_i, \hat{y}_i^{(t-1)}\right)$ and $h_i = \partial^2_{\hat{y}_i^{(t-1)}} \mathrm{L}\left(y_i, \hat{y}_i^{(t-1)}\right)$

Removing the constant terms since they don't affect the optimization, the objective function at the t-th step is derived below. The goal is to find a f_t to optimize $Obj^{(t)}$.

$$Obj^{(t)} = \sum_{i=1}^{N}[\, g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t)$$

8. Finding a tree in each step to improve the prediction along the gradient is critical in XGB. For a decision tree, internal node defines the data point flowing direction. Each leaf is assigned a weight, which is the prediction. Mathematically, a tree can be defined as $f_t(x) = w_{q(x)}$, where $q(x)$ is a directing function that assigns every data point to the $q(x)$-th leaf. $w_{q(x)}$ is the corresponding score on the $q(x)$-th leaf. An index set is also defined as $I_j = \{i | q(x_i) = j\}$. It contains the indices of data points that are assigned to the j-th leaf. Rewriting the objectives in terms of leaves, the objective function becomes

$$Obj^{(t)} = \sum_{i=1}^{N}[\, g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \mathrm{T} + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

$$Obj^{(t)} = \sum_{j=1}^{T}[(\sum_{i \in I_j} g_i) w_j + \frac{1}{2}(\sum_{i \in I_j} h_i + \lambda) w_j^2] + T$$

The objective function in this form would be optimized by w_j *and* w_j^2. The best w_j that optimizes the objective function is $w_j^{best} = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda}$, and the corresponding objective function is

$$Obj^{(t)} = -\frac{1}{2}\sum_{j=1}^{T}\frac{\left(\sum_{i\in I_j} g_i\right)^2}{\sum_{i\in I_j} h_i + \lambda} + T$$

9. When building a tree, to find the best splitting point that can optimize the objective function, the best splitting point of each attribute is identified first, then the best attribute is picked out based on the objective function. Since I is the set of indices of data points that assigned to a node, I_L and I_R are the sets of indices of data points that assigned to two new leaves. The gain of splitting is calculated based on optimal objective function. The split that achieves the most gain is the best one.

$$gain = \frac{1}{2}\left[\frac{\left(\sum_{i\in I_L} g_i\right)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{\left(\sum_{i\in I_R} g_i\right)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{\left(\sum_{i\in I} g_i\right)^2}{\sum_{i\in I} h_i + \lambda}\right] - r$$

r is the complexity cost by introducing additional leaf. The tree is built to the maximum depth in this way, and is pruned by taking out the nodes with negative gains in a bottom-up order.

10. When building an individual tree, a subset of instances N is sampled. At each split, a subset of attributes M is also randomly selected. A XGB model can be created by following steps in 1 through 9.

11. We constructed ten XGB models as base models in this research. Setting up seed was tried randomly to ensure a repeatable predictive result for each XGBoost model. However, it didn't work and couldn't provide s repeatable prediction. Parameter "eta" in the R XGBoost package was adjusted from 0.1 to 1 to control the gradient speed. They are listed in table 3 for reference. Parameter "nround" was optimally chosen by a 10-fold cross validation method based on parameter "eta". As a result, the number of individual trees in each XGB model was different,

therefore these ten XGB models had different structures and provided a different prediction on testing data set.

Table 3

Parameter Eta in Extreme Gradient Boosting Base Model

Extreme Gradient Boosting	Eta
1	0.1
2	0.2
3	0.3
4	0.4
5	0.5
6	0.6
7	0.7
8	0.8
9	0.9
10	1.0

Calculating factor scores of multiple correspondence analysis

Multiple correspondence analysis (MCA) was applied to the outputs of the base models (Le Roux & Rouanet, 2004; Greenacre & Blasius, 2006). Factor scores of individual output instances were produced (Abdi & Valentin, 2007). They were added as new attributes to ensemble with the outputs of base models in the final ensemble step (Zhang & Zhang, 2009). MCA is a statistical procedure that applied to categorical variables, which represents data in a low-dimensional Euclidean space. In our study, conducting MCA converted the B outputs of base models into a set of factor scores.

Since the dissertation is focused on classification problems, we kept the B outputs of base models in categorical format. Each output must be reconstructed into

another set of binary variables with only 0 and 1 as their values. For example, for a binary output Salary with two categories "more than $50,000" and "not more than $50,000", two new variables "more than $50,000" and "not more than $50,000" are created to replace the categorical output. For a person with salary more than $50,000, the corresponding value of the new variable "more than $50,000" is 1, and "not more than $50,000" is 0. In this way, each categorical output with J_k levels is replaced by J_k new binary variables. With B outputs in total, J new binary variables were created and set into the MCA approach. For I observations, an indicator matrix $\boldsymbol{X}$ with J columns and I rows was formed.

A correspondence analysis (CA) was then performed on the indicator matrix. Letting N denote the sum of elements of indicator matrix, the probability matrix $\mathbf{Z}$ is $\mathbf{Z} = N^{-1}\mathbf{X}$. The vector of row sums of $\mathbf{Z}$ is denoted as $\mathbf{r}$. The vector of column sums of $\boldsymbol{Z}$ is denoted as $\mathbf{c}$. The following singular value decomposition is performed.

$$\mathbf{D_r}^{-\frac{1}{2}}(\mathbf{Z} - \mathbf{r}\mathbf{c}^{\mathrm{T}})\mathbf{D_c}^{-\frac{1}{2}} = \mathbf{P}\Delta\mathbf{Q}^{\mathrm{T}}$$

where $\mathbf{D_c} = \mathrm{diag}\{\mathbf{c}\}$, $\mathbf{D_r} = \mathrm{diag}\{\mathbf{r}\}$, Δ is the diagonal matrix of the singular values which is calculated from Δ^2, the matrix of eigenvalues. Row and column factor scores, $\mathbf{F}$ and $\mathbf{G}$, are calculated as follows:

$$\mathbf{F} = \mathbf{D_r}^{-\frac{1}{2}}\mathbf{P}\Delta$$

$$\mathbf{G} = \mathbf{D_c}^{-\frac{1}{2}}\mathbf{Q}\Delta$$

These factors scores are considered as inheriting the maximum possible variance from $\mathbf{X}$. Although MCA produces row and column factor scores, in our approach, only the

column factor scores were ensembled with the outputs of base models in the last ensemble step.

Choosing optimal subsets of base models

Choosing optimal subsets of base models is the third step in the whole procedure. Various model selection techniques could be applied to choose optimal subsets of base models to ensemble. The following two methods were used in this research for model selection.

Cramér's V correlation analysis: Cramér's V correlation between outputs of base models on testing data is calculated. Then, a criterion or a cutout point of correlation coefficient is picked, and the most uncorrelated models are chosen as the group of optimal base models (Cramér, 1946).

For B outputs of base models $\{\hat{y}_1, \hat{y}_2, \dots, \hat{y}_B\}$, Cramér's V correlation coefficient measures the pairwise association between them. The association is based on Pearson's chi-square statistics. B ranges from 1 to 31 since thirty-one base models were generated in total. Cramér's V correlation coefficient is calculated based on the following formula. For two outputs of base model, $\hat{y}_i$ and $\hat{y}_j$, $i \neq j, \forall i = 1, \dots, B$, and $\forall j = 1, \dots, B$, a contingency table is created in table 4, k is the number of classes of the output variable.

Table 4

Contingency Table of Cramér's V Correlation

Category of Output	$\hat{y}_i$	$\hat{y}_j$
1	$n_{1,i}$	$n_{1,j}$
2	$n_{2,i}$	$n_{2,j}$
.	.	.
.	.	.
.	.	.
k	$n_{k,i}$	$n_{k,j}$

$n_{k,i}$ is the number of class k observed in the output $\hat{y}_i$. $n_{k,j}$ is the number of class k observed in the output $\hat{y}_j$. The chi-squared statistic is calculated as below

$$\chi^2 = \sum_k \frac{(n_{k,i} - n_{k,j})^2}{n_{k,j}}$$

Cramér's V correlation coefficient is

$$V = \sqrt{\frac{\chi^2/N}{k-1}} = \sqrt{\frac{\varphi^2}{k-1}}$$

where N is the grand total of observations, $\varphi^2$is the phi coefficient.

Cramér's V correlation coefficient ranges from 0 to 1. A value close to 0 indicates less correlation between two outputs. Since the base models are expected to be accurate, which leads their pairwise correlation coefficients closer to 1. The cutout point value of its absolute value varies based on different data sets. We chose base model pairs whose correlation coefficient was closer to 0 as members of the optimal subset.

<u>Backward selection</u>: A backward selection of base models $\hat{Y}$ was applied in the research based on Akaike Information Criterion (AIC) value provided by the logistic regression ensemble models. AIC was originally introduced to measure the relative

quality of models on the same set of data set (Hocking, 1976; Bozdogan, 1987). It is defined as follows,

$$AIC = 2k - 2\ln(\hat{L})$$

where k is the number of estimated parameters in a model, $\hat{L}$ is the maximum value of the likelihood function of logistic regression model. In this research, the data set contains the output of base models $\hat{Y}$. For B outputs of base models $\{\hat{y}_1, \hat{y}_2, \dots, \hat{y}_B\}$, this approach starts from ensemble all of the outputs of base models by logistic regression model below.

$$\pi(\hat{Y}) = \frac{e^{g(\hat{Y})}}{1 + e^{g(\hat{Y})}}$$

Where $g(\hat{Y}) = \beta_0 + \beta_1\hat{y}_1 + \cdots + \beta_B\hat{y}_B$, and $\beta_0, \beta_1, \dots, \beta_B$ are the fitted coefficients of logistic regression.

In this research, the backward selection method excluded one base model in each round to reach the goal of not significantly losing predictive power with the smallest number of base models. If B denotes the number of base models, in each round, AICs of B number of logistic regressions were compared. Here, each logistic regression was created by combining $B - 1$ number of base models by omitting one base model. Each base model was excluded once in a logistic regression. Thus, the resulted AIC value of logistic regression presented the effect of each base model to the predictive power. One base model was chosen to exclude in the next round if omitting it resulted in the smallest AIC value. The backward selection stopped if excluding any one of the remaining base models would not make the AIC significantly lower than that in the previous round. Here, the chi-square χ^2 statistic was applied to determine the significance of AIC decreasing at 0.05 level in the study.

Integrating base models

In the last step of the ensemble approach, majority voting, random forest, extremely gradient boosting, and logistic regression models were applied to integrate the outputs of base models with 1) all base models, 2) all base models and factor scores of multiple correspondence analysis, 3) the optimal subsets of base models chosen by Cramér's V and backward model selection, or 4) factor scores and the optimal subsets of base model chosen by Cramér's V correlation and backward model selection. The misclassification rate was used to compare all of the ensemble results.

Software and Code

Experiments were conducted in RStudio of R version 3.3.1 (R Core Team, 2016). RStudio is an integrated development environment (IDE) for R. Compared to R, RStudio is designed to be more user-friendly. Researchers can code, edit, and run R codes in RStudio. The open-sourced RStudio is available to download for free and is used in this research. The RStudio for windows desktop was chosen and downloaded from the following website, https://www.rstudio.com, by selecting platform x86_64-w64-mingw32/x64 (64-bit). A screen shot of RStudio interface can be found in Appendix A.

Since many researchers contribute their research results to the R community for free, the R community is the first or best place for a researcher to find solutions to classification problems. Random forest, extremely randomized trees (extra trees), and extreme gradient boosting model are all available in the R community, so R becomes an accessible option to conduct experiments in this dissertation. In addition to the

models mentioned above, multiple correspondence analysis and other well-known statistical analyses are all available in the R community. Data manipulation and calculation are also convenient to conduct in R. Due to the easy access to the R community and documentation and tutorial of R programming, R code was used in RStudio for all the experiments in this research.

In addition to basic R programming, research ideas are contributed to the R community and presented by researchers in R packages. The three types of models, random forest, extremely randomized trees, and extreme gradient boosting, are presented in three R packages, XGBoost, extraTrees, and randomForest. They have been widely used by many researchers in their research (Chen, 2014; Chen & He, 2015; Diaz-Uriarte, & Alvarez de Andres, 2006; Geurts et al., 2006). The manual of all R packages, related R code, and examples are saved in the Comprehensive R Archive Network (CRAN) and maintained regularly by the authors. CRAN can be accessed at https://cran.r-project.org/. Detailed information of R packages of random forest, extremely randomized trees, extreme gradient boosting, and logistic regression model are listed in table 5. The syntax of conducting models in R code can be found through the links provided in the reference list.

Table 5

R Packages of Models

Model	R Package	Reference
Extreme Gradient Boosting	xgboost	Chen , He , & Benesty, 2016
Extremely Randomized Trees	extraTrees	Simm, & Magrans de Abril, 2014
Random Forest	randomForest	Breiman, Cutler, Liaw, & Wiener, 2015
Logistic Regression	glm	Simon, 1992

In addition to R packages which generate the above base models or ensemble models, multiple correspondence analysis and Cramér's V correlation analysis were also applied in the experiments in RStudio. The related R packages, ca and vcd, are listed in table 6.

Table 6

R Packages of Analysis

Analysis	R Package	Reference
Multiple Correspondence	ca	Greenacre, Nenadic, & Friendly, 2016
Cramér's V correlation	vcd	David, Achim, Kurt, Florian, & Michael, 2016

Several other R packages, which supported models and analysis in the research, are listed in table 7. They were used for data manipulation and calculation, such as installing R packages, binarizing predictors, supporting extra tree package, and calculating variable importance when building a model, selecting base models, and integrating base models.

Table 7

Supportive R Packages

R Package	Function	Reference
caret	Data manipulation	Kuhn et al., 2016
DiagrammeR	Plot variable importance	Iannone, 2016
Ckmeans.1d.dp	Plot variable importance	Song & Wang, 2016
rJava	Support Extra Tree package	Urbanek, 2016
drat	Install R packages	Eddelbuettel et al., 2016

Data Sets

The research experiments were conducted on three UCI data sets that are public and free to download (Lichman, 2013; Blake & Merz, 1998). UCI is a repository of machine learning database. These data sets are real-world data and extensively used by researchers in many research studies (Ron, 1996; Yeh & Lien, 2009; Thuraisingham, Tran, Boord & Craig, 2007). These three data sets represent binary classification problems with different class ratios of the target attribute. They were used to test if the proposed ensemble approach achieved better classification accuracy on binary classification problems. The profiles of the three data sets are listed in table 8.

Table 8

Data Sets

Data Set	# of Attributes	# of Instances	Class of Target	Class Ratio of Target Attribute
Adult	14	48842	2	24% vs 76%
Credit Card Clients	23	30000	2	22% vs 78%
EEG Eye State	14	14980	2	45% vs 55%

Note: # means count

The Adult data set is provided by UCI as two separate sets: training and testing data sets. The Credit Card Clients and EEG Eye State data sets were partitioned into training and testing data sets in a 70% vs. 30% ratio. Base models were built on training data sets. Prediction was provided by base models on testing data sets. The number of instances in training and testing data sets are listed in table 9 as follows.

Table 9

Training and Testing Data Sets

Data Set	Number of Instances in Training Set	Number of Instances in Testing Set
Adult	32561	16281
Credit Card Clients	21000	9000
EEG Eye State	10486	4494

Experiment Design

There are four experiment designs. They were designed to explore how model selection, MCA factor scores, and ensemble method affected the classification accuracy. They were also designed to identify ensemble strategies to improve the ensemble performance. Detailed designs and what research questions were answered are presented and explained below.

1. Ensemble all base models
2. Ensemble all base models and MCA factor scores
3. Ensemble with backward or Cramér's V model selection
4. Ensemble with MCA factor scores and backward or Cramér's V model selection

Ensemble all Base Models

This experiment was performed in the following steps. Ten base models of random forest (RF), eleven extremely randomized trees (ERT), and ten extreme gradient boosting (XGB) models were first generated. Then, majority voting, random forest, and extreme gradient boosting model were applied to ensemble all of those

thirty-one base models. The only reason for creating eleven extremely randomized trees is to avoid even voting of the binary classification of target in the majority voting ensemble. There is also no specific reason for choosing extremely randomized trees as the thirty-first base model.

The average accuracy of ten random forest base models was considered as the benchmark in our research because of its well-known reputation of good performance. The accuracy of ensemble results was compared with that of each base model and the benchmark to find out whether the four ensemble methods helped to increase the accuracy. The accuracy of ensemble results was also compared with each other. The best ensemble method among random forest, extreme gradient boosting, logistic regression and majority voting was identified when integrating the thirty-one base models.

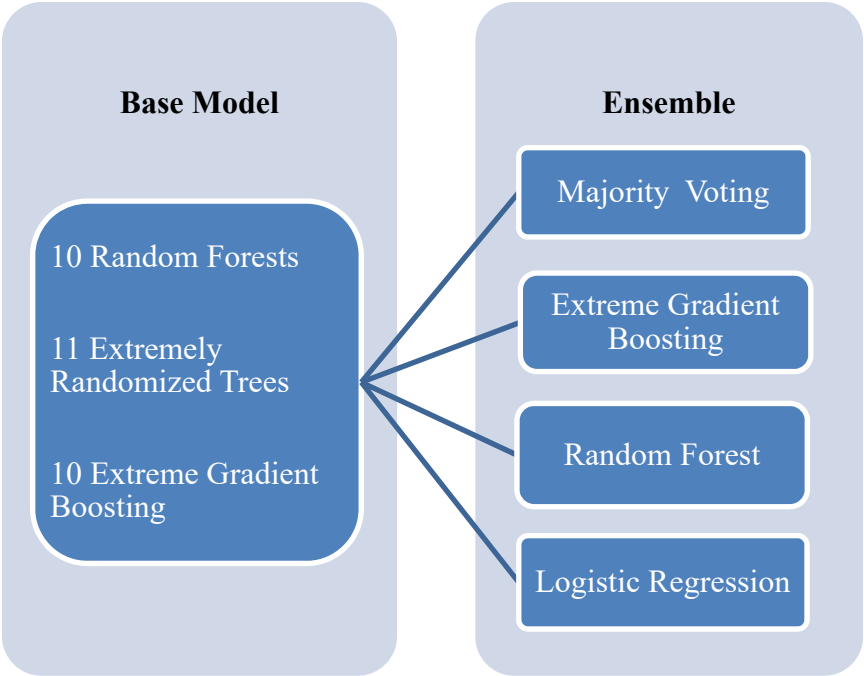


Figure 1. Ensemble all Base Models

The experiment answered the following research questions.

1. Will the four ensemble approaches of ensemble-based models increase the predictive accuracy when compared with the benchmark or the individual ensemble models?
2. As base classifiers, are random forest, extremely randomized trees, and extreme gradient boosting models good candidates to be ensembled?
3. How will various model combinations (majority voting, random forest, extreme gradient boosting, and logistic regression) affect the predictive accuracy of the ensemble approach?

Ensemble all Base Models and MCA Factor Scores

In addition to the experiment that integrated all base models, the experiment in this section was performed by adding factor scores of multiple correspondence analysis. Multiple correspondence analysis was applied to the predictions of thirty-one base models to generate MCA factor scores. MCA factor scores were considered to represent the maximum variance of the thirty-one base models. Because of the different nature of individual data sets, a different number of sets of factor scores were generated for the three data sets used in the experiment. The number ranged from 4 to 6. Random forest, extreme gradient boosting, and logistic regression model were applied to ensemble all the base models and the factor scores of multiple correspondence analysis. In this experiment, the base models are the same as those in the first experiment design. Majority voting is not applicable in the experiment because MCA factor scores were numerical but not categorical attribute for ensemble.

The performance of ensemble results was compared with those of each base model and the benchmark to determine whether the four ensemble methods increased the predictive accuracy. Compared with the experiment that only integrated base models, the factor scores of multiple correspondence analysis were added as predictors in the final ensemble to identify whether adding MCA factor scores increased the predictive accuracy.

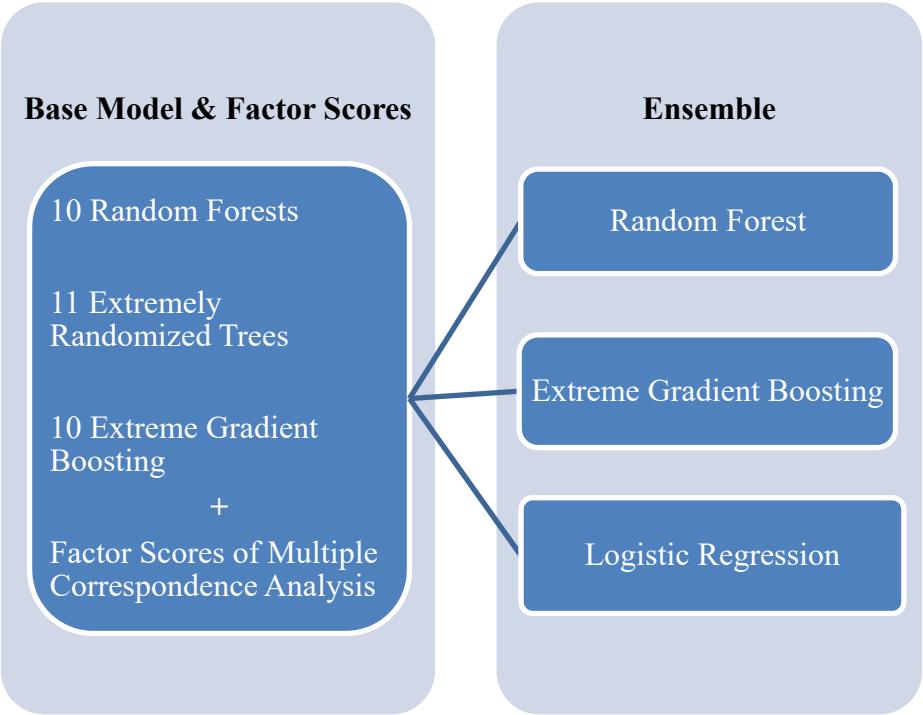


Figure 2. Ensemble all Base Models and MCA Factor Scores

The three ensemble results were compared with each other. The one with the best performance among random forest, extreme gradient boosting, and logistic regression was determined when combining the thirty-one base models and MCA factor scores.

The experiment answered the following research questions.

1. Will the three ensemble approaches of ensemble-based models increase the predictive accuracy when compared with the benchmark or the individual ensemble models?
2. As base classifiers, are random forest, extremely randomized trees, and extreme gradient boosting model good candidates to be ensembled with MCA factor scores?
3. Will the multiple correspondence analysis make a difference on the predictive accuracy of the overall ensemble approach?
4. How will various model combinations (random forest, extreme gradient boosting, and logistic regression) affect the predictive accuracy of the ensemble approach?

Ensemble with Cramér's V correlation or Backward Model Selection

Compared with the second experiment design, the model selection procedure was added in, but factor scores of multiple correspondence analysis was excluded from the experiment presented in this section. Two methods, Cramér's V correlation or Backward Model Selection, were applied in the model selection step. The whole procedure includes three steps: base model generation, model selection, and ensemble. The base models generated in the first step are the same as those in the previous experiment designs.

The first model selection method is derived from Cramér's V correlation analysis. In this method, Cramér's V correlation coefficient is calculated between each pair of base models. Paired base models with correlation coefficient lower than a threshold value are kept in the final ensemble step. In order to keep the diversity of

base models, base models which are less correlated with each other were kept. The final ensemble method was applied to the selected base models by majority voting, random forest, and extreme gradient boosting model. Since the threshold of Cramér's V correlation coefficient value was hard to determine, we didn't use individual correlation coefficient as the cutout point to pick the least related base models. We evaluated the average value of Cramér's V correlation coefficient between different types of base models to select two types of least correlated base models. The base models in those two types were selected and kept in the final ensemble procedure. The Cramér's V correlation coefficients of each paired base models on the three data sets are presented in Appendix C, D and E. It is shown that extreme gradient boosting and extremely randomized trees base models have the smallest average Cramér's V correlation coefficient for the three data sets. Therefore, all extreme gradient boosting and extremely randomized trees base models were selected as the optimal base models and combined in the final ensemble step.

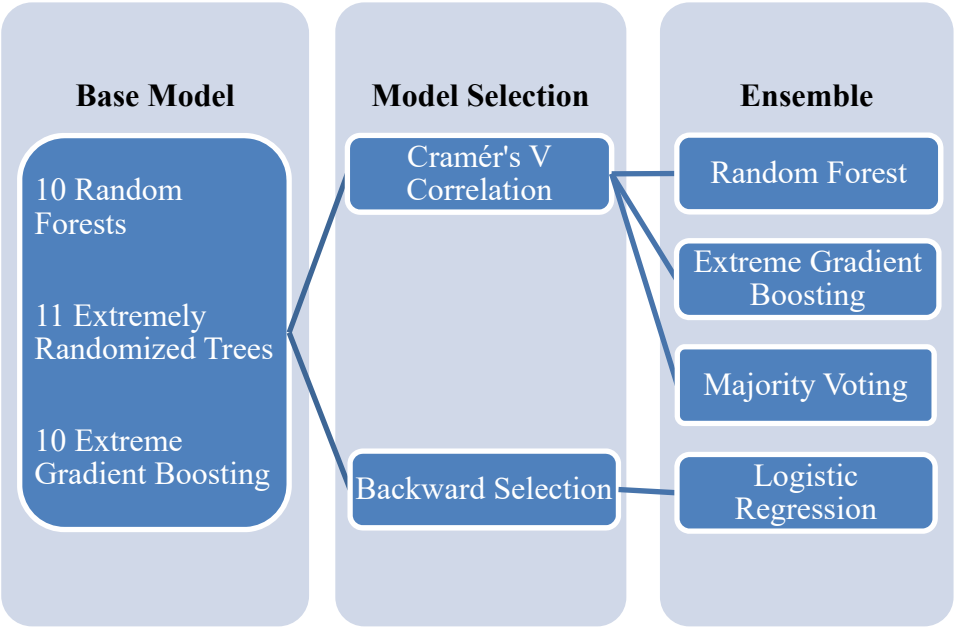


Figure 3. Ensemble with Model Selection

Another model selection method is backward selection based on Akaike's information criterion (AIC) which is combined with logistic regression model. This model selection procedure starts from ensemble all base models. Then, base models are removed one by one in the ensemble step until the AIC value doesn't decrease significantly at 0.05 level. In each round, the base model that contributed to the AIC the most is chosen and excluded in the next round.

The accuracy of ensemble results was compared with those of each base model and the benchmark to determine whether the four ensemble methods with model selection was able to increase the predictive accuracy. The best candidate base models chosen by different model selection methods were selected. Compared with the experiment design which integrated all base models without model selection, we added the model selection procedure in the experiment presented in this section. By comparing the performance of those two experiments, we were able to find out whether the model selection procedure can help to increase the predictive accuracy. Lower predictive accuracy might be observed because a smaller number of base models, which meant less information, were used in the final ensemble. Whether model selection helped on combining tree-based ensemble models was also learned by comparing the ensemble performance in experiment two.

The accuracy of ensemble results of the two model selection methods was compared to find out which one was the better model selection method. We also identified which combination of model selection method and final ensemble method worked the best together in increasing predictive accuracy.

The experiment answered the following research questions.

1. Will the four ensemble approaches of ensemble-based models increase the predictive accuracy when compared with the benchmark or the individual ensemble models?
2. Are random forest, extremely randomized trees, and extreme gradient boosting good candidates as base classifiers when applying model selection in ensemble?
3. How will various model combinations (random forest, extreme gradient boosting, and logistic regression) affect the predictive accuracy of the ensemble approach?
4. Will the two types of model selections make a difference in the predictive accuracy of the overall ensemble approach?

Ensemble with MCA Factor Scores and Model Selections

In this experiment design, factor scores of multiple correspondence analysis were added into the experiment. The whole procedure involved three steps, base model and factor score creation, base model selection, and ensemble. In the first step, ten random forest, eleven extreme randomized trees, and ten extreme gradient boosting base models were created. They were the same base models as those in the previous three experiment designs. Then, multiple correspondence analysis was applied to the predictions of the base models to generate the factor scores of MCA. The factor scores of MCA were also the same as those in the second experiment design. MCA factor scores were then integrated together with the selected base models in the final model ensemble step by logistic regression, random forest, and extreme gradient boosting model. Since majority voting ensemble can only be applied to the outputs of base

models, but MCA factor scores were not outputs of base models, majority voting ensemble was not applicable in the experiment.

The two model selection methods, backward AIC selection and Cramér's V correlation selection, were used in this experiment. The backward selection method started from combining all base models and MCA factor scores. Model selection and ensemble worked together to evaluate base models and MCA factor scores one by one, and then determine which base model or factor scores contributed the most AIC that provided by logistic regression ensemble in each round. The identified base model or factor scores would be excluded in the next round of evaluation. Only one base model or one factor score was eliminated in each round. The AIC value of logistic regression model in each round was compared with that in the previous round. If the AIC value didn't decrease significantly at 0.05 level, the backward selection stopped. The experiment showed that the same group of base models was selected as in experiment three. Figure 4 shows the experiment structure of logistic regression ensemble with backward selection.

The accuracy of the ensemble result was compared with the accuracy of each base model and the benchmark. Whether integrating the backward model selection and factor scores increased the predictive accuracy was determined. Compared with the ensemble method without factor scores but with backward model selection, whether the method that integrated the factor scores with backward selection increased the predictive accuracy was learned.

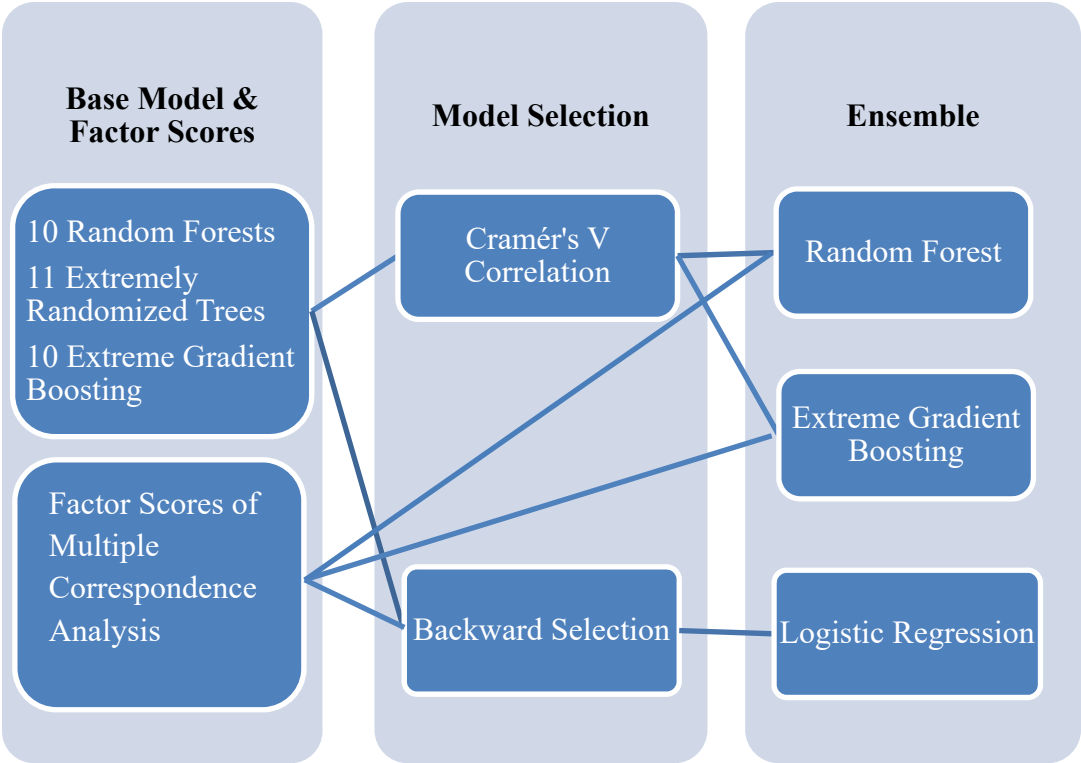


Figure 4. Ensemble with Factor Scores and Model Selection

The second model selection method is applying Cramér's V correlation analysis to select the least correlated base models. Cramér's V correlation coefficient is calculated between each base model. Ideally, paired base models with a correlation coefficient lower than a threshold value are kept to ensemble with factor scores in the final ensemble step. However, the correlations between base models on different data sets vary, thus the threshold is not easy to determine. We decided to evaluate the average correlation coefficients of different types of base models. Two types of base models that had the lowest average correlation were chosen. Then all base models in these two types were combined in the final ensemble. The selection procedure and the selected base models are the same as those in experiment three. It was shown that extreme gradient boosting and extremely randomized trees base models had the

smallest average Cramér's V correlation coefficient for the three UCI data sets. Therefore, all base models in these two types were selected in the final ensemble step. The final ensemble methods for the factor scores and the selected base models are random forest and extreme gradient boosting model.

We compared the accuracy of ensemble results with the accuracy of individual base model and the benchmark. Whether the two ensemble methods with Cramér's V model selection and factor scores increased the predictive accuracy, and which ensemble method performed the best were determined. Compared with the ensemble method without factor scores but with Cramér's V model selection, whether integrating the factor scores with selected base models increased the predictive accuracy was also determined. From the experiment result, the best candidate of base models chosen by different model selection methods was learned.

We also compared the accuracy of ensemble results of the two different types of model selection methods. The better model selection method among backward selection and Cramér's V selection was determined when integrating factor scores of multiple correspondence analysis in the final ensemble step. The combination of model selection and final ensemble method which helps increase predictive accuracy the most was also learned.

The experiment answered the following research questions.

1. Will the three ensemble approaches of ensemble-based models increase the predictive accuracy when compared with the benchmark or the individual ensemble models?

2. Are random forest, extremely randomized trees, and extreme gradient boosting good candidates as base classifiers when ensembling with factor scores of multiple correspondence analysis and applying model selection?
3. Will the multiple correspondence analysis make a difference on the predictive accuracy of the overall ensemble approach when integrating two different types of model selection?
4. How will various model combination methods affect the predictive accuracy of the ensemble approach?

Summary

To achieve the dissertation goal and answer the research questions, three data sets from an open source server, UCI, were used to test our research ideas. Various experiments were conducted using R code in RStudio. Through the experiments, whether integrating ensemble-based models increased predictive accuracy, how different ensemble-based models worked when they were further ensembled, and how multiple correspondence analysis performed in the ensemble was studied. Experiments with four different designs were conducted in the research. Experiment results between different designs were compared. How multiple correspondence analysis and base model selection affected the ensemble approach was studied.

Thirty-one base models were generated: ten random forest models, ten extreme gradient boosting models, and eleven extremely randomized tree models. These base models were the same in all four designs. In the first experiment, all base models were integrated by majority voting, random forest, extreme gradient boosting, and logistic regression. In the second experiment, all base models were combined with factor scores

of multiple correspondence analysis by the same three ensemble methods in experiment one excluding majority voting. In the third experiment, factor scores were excluded but two base model selection methods were added. When applying Cramér's V model selection method, the ensemble methods were the same three ones used in experiment one excluding logistic regression. When applying backward model selection method, the ensemble model was logistic regression. The fourth experiment has the same ensemble structure as in the third experiment, however factor scores of multiple correspondence analysis was integrated in the ensemble approach. Part of the fourth experiment utilized backward model selection and logistic regression ensemble method. The rest of the fourth experiment adopted Cramér's V base model selection with two ensemble methods, random forest and extreme gradient boosting. Those four experiments were designed to answer our research questions in different situations step by step. Table 10 summarizes the structure of the four experiment designs. Appendix B shows the R code for all the four experiment designs for data set EEG as an example.

Table 10

Structure of Experiment Designs

Experiment Design	Ensemble Variables	Model Selection	Ensemble Methods
One	all base models	none	Majority Voting, Extreme Gradient Boosting, Random Forest, Logistic Regression
Two	all base models + MCA factor scores	none	Extreme Gradient Boosting, Random Forest, Logistic Regression
Three	selected base models	Backward or Cramér's V	Majority Voting, Extreme Gradient Boosting, Random Forest, Logistic Regression

Experiment Design	Ensemble Variables	Model Selection	Ensemble Methods
Four	selected base models + MCA factor scores	Backward or Cramér's V	Extreme Gradient Boosting, Random Forest, Logistic Regression

Chapter 4

Results

Base Models

The first step of all four experiments was generating ten extreme gradient boosting (XGB), eleven extremely randomized trees (ERT), and ten random forest (RF) models. The same thirty-one models were generated as base models in all the four experiments. Training data sets were used to generate base models, which then provided predictions for testing data sets. All the classification accuracies reported in this research are based on testing data sets. The classification accuracies of each base model on the three UCI data sets are summarized in table 11.

Table 11

Classification Accuracy of Base Models

Data Set	Base Model	RF		ERT		XGB	
		# of Trees	Accuracy	# of Trees	Accuracy	Eta	Accuracy
Adult	1	50	0.8644	50	0.8450	0.1	0.8706
	2	100	0.8649	100	0.8433	0.2	0.8728
	3	150	0.8640	150	0.8446	0.3	0.8708
	4	200	0.8646	200	0.8452	0.4	0.8730
	5	250	0.8651	250	0.8452	0.5	0.8745
	6	300	0.8642	300	0.8445	0.6	0.8762
	7	350	0.8649	350	0.8455	0.7	0.8751

Data Set	Base Model	RF		ERT		XGB	
		# of Trees	Accuracy	# of Trees	Accuracy	Eta	Accuracy
	8	400	0.8649	400	0.8458	0.8	0.8770
	9	450	0.8642	450	0.8450	0.9	0.8767
	10	500	0.8651	500	0.8458	1.0	0.8755
	11	N/A	N/A	550	0.8456	N/A	N/A
	Average	N/A	0.8646	N/A	0.8450	N/A	0.8742
Credit Card Clients	1	50	0.8143	50	0.8121	0.1	0.8169
	2	100	0.8170	100	0.8143	0.2	0.8203
	3	150	0.8176	150	0.8134	0.3	0.8234
	4	200	0.8174	200	0.8130	0.4	0.8236
	5	250	0.8172	250	0.8130	0.5	0.8254
	6	300	0.8156	300	0.8130	0.6	0.8262
	7	350	0.8150	350	0.8130	0.7	0.8257
	8	400	0.8176	400	0.8148	0.8	0.8258
	9	450	0.8187	450	0.8148	0.9	0.8260
	10	500	0.8169	500	0.8148	1.0	0.8256
	11	N/A	N/A	550	0.8148	N/A	N/A
	Average	N/A	0.8167	N/A	0.8137	N/A	0.8239
EEG Eye State	1	50	0.9243	50	0.9372	0.1	0.9009
	2	100	0.9268	100	0.9424	0.2	0.9059
	3	150	0.9237	150	0.9446	0.3	0.9003
	4	200	0.9295	200	0.9455	0.4	0.9119
	5	250	0.9292	250	0.9439	0.5	0.9105
	6	300	0.9308	300	0.9468	0.6	0.9089
	7	350	0.9288	350	0.9435	0.7	0.9061
	8	400	0.9306	400	0.9453	0.8	0.8988
	9	450	0.9299	450	0.9450	0.9	0.8925
	10	500	0.9288	500	0.9473	1.0	0.8636
	11	N/A	N/A	550	0.9473	N/A	N/A
	Average	N/A	0.9282	N/A	0.9444	N/A	0.8999

Extreme gradient boosting base models provided better average or individual classification accuracy than random forest and extremely randomized trees for the Adult and Credit Card Clients data sets. However, it provided lower average or

individual classification accuracy than the other two types of base models on the EEG State data set. Extremely randomized trees provided lower individual and average classification accuracy than the other two types of models on the Adult and Credit Card Clients data sets. However, it provided better individual and average classification accuracy on the EEG State data set than the other two types of base models. Random forest base models had predictive accuracy that ranged between those produced by the other two types of base models on all three data sets. The difference of classification accuracy between the best and worst base model on the three data sets ranged from 1.7% to 9.7%, which is shown in table 12. The table also lists the base models that performed the best and the worst on different data sets.

Table 12

Best and Worst Classification Accuracy of Base Models

Data Set	Base Model		Accuracy Difference (%)
	Best Accuracy	Worst Accuracy	
Adult	XGB (0.8770)	ERT (0.8433)	4.0%
Credit Card Clients	XGB (0.8262)	ERT (0.8121)	1.7%
EEG Eye State	ERT (0.9473)	XGB (0.8636)	9.7%

Random forest base models were generated by setting up random seeds and different numbers of individual trees in each forest. The reported classification accuracy is replicable by using the same seed value and tree number. Figure 5 shows that there is no linear trend of classification accuracy associated with the number of trees in the random forest for the three data sets. A forest with more individual trees doesn’t guarantee a better classification accuracy.

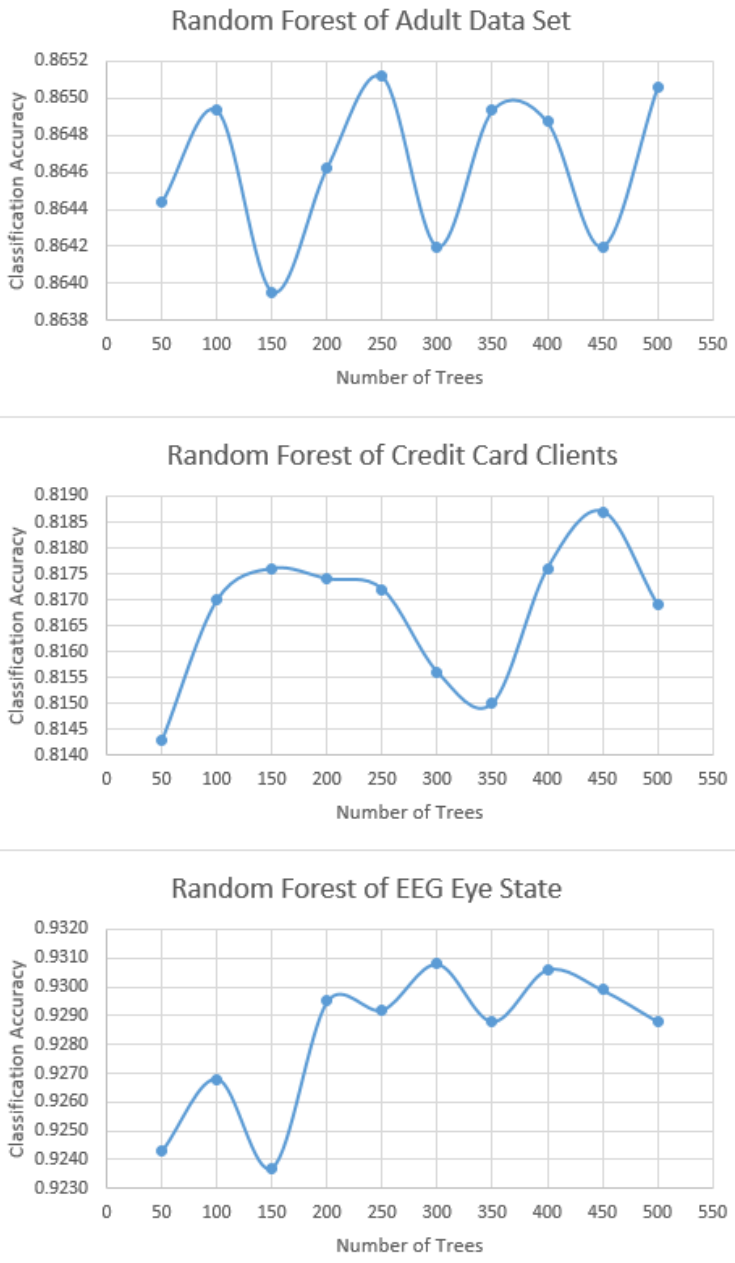


Figure 5. Classification Accuracy of Random Forest Base Model

Extremely randomized trees base models were also generated by setting up random seeds and different numbers of individual trees. The reported classification accuracies are also replicable when setting up the same seed and the same number of

individual trees. Figure 6 shows that the predictive accuracy is not linearly associated with the number of individual trees. Including a larger number of individual trees in a model does not provide a better classification accuracy.

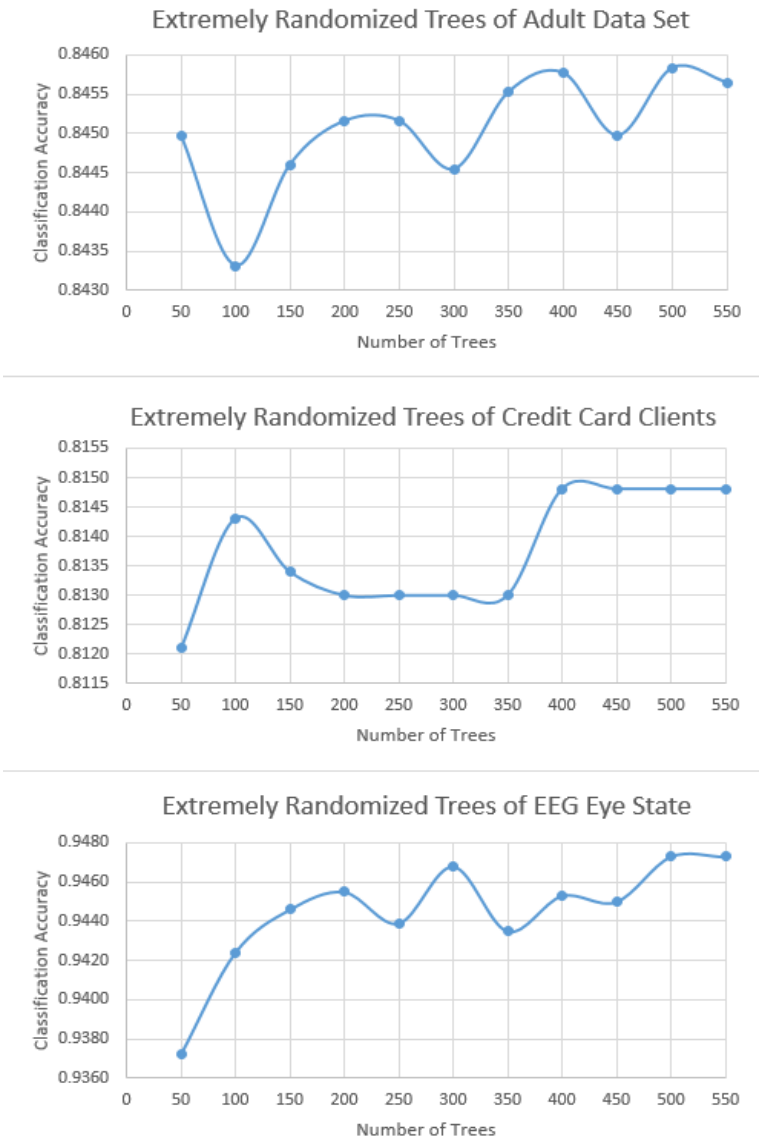


Figure 6. Classification Accuracy of Extremely Randomized Trees Base Model

The extreme gradient boosting base models were generated by adjusting the parameter “eta” in the xgboost R package, and then identifying the optimal number of

trees with a ten-fold cross validation method for the specific “eta” setting to achieve the best predictive accuracy. Eta is defined and used in the xgboost R package to adjust the gradient pace of boosting, thus generating different XGB models. In our experiment, we applied different value of “eta” to generate ten different XGB base models. Because of the randomization nature of the model, they are not replicable even with the same setting up of parameters or random seeds. Figure 7 shows that there is no consistent linear trend between classification accuracy and parameter “eta”. Smaller “eta” doesn’t guarantee better performance.

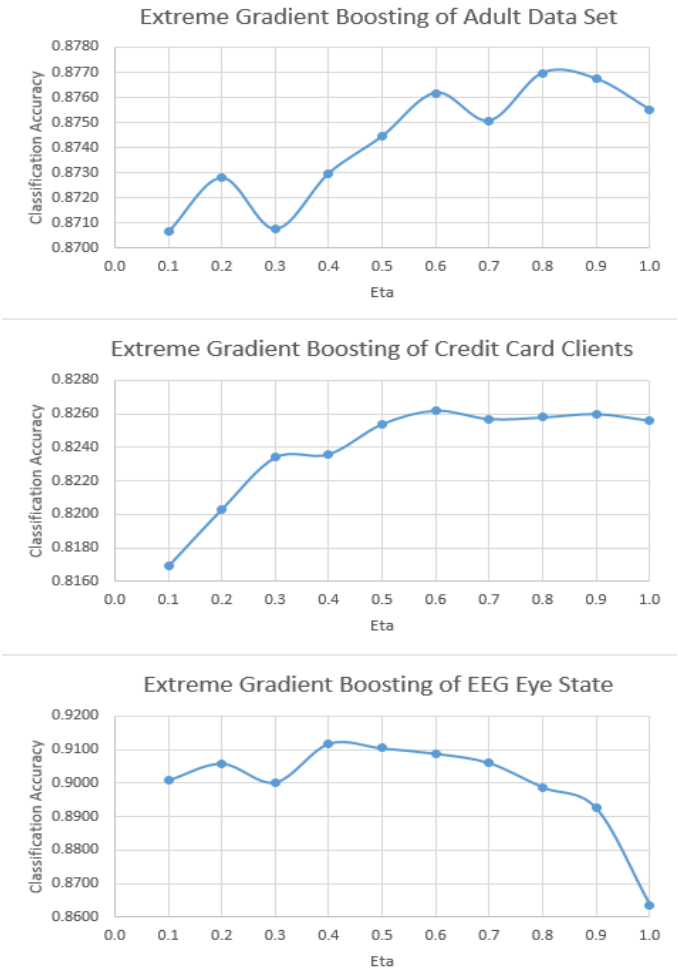


Figure 7. Classification Accuracy of Extreme Gradient Boosting Base Model

As mentioned in the research methodology, the performance of the random forest base model was set up as benchmarks for the three data sets. Although random forest performed not the worst or best on providing classification accuracy on average when compared with the other two types of base models, the average classification accuracy of random forest base models was still set up as a benchmark value for each data set as proposed in our research. They were compared with the ensemble classification accuracy and helped to answer our research questions. The benchmark values for the three data sets are listed below in table 13.

Table 13

Benchmarks of Classification Accuracy

Data Set	Benchmark Classification Accuracy
Adult	0.8646
Credit Card Clients	0.8167
EEG Eye State	0.9282

Multiple Correspondence Analysis

Multiple correspondence analysis (MCA) was applied to the predictions of thirty-one base models. It was conducted to capture the maximum variance of base model predictions in a low-dimensional Euclidean space. In other words, it represents integrated features of base model outputs. Factor scores were generated as the results of multiple correspondence analysis. Although MCA produced row and column factor scores, in our experiment, only the column factor scores were chosen and combined with the outputs of base models in the last step. Since different data sets present different natures, the number of generated factor scores on different data sets varied

from 4 to 6 sets. Table 14 shows that based on the prediction of thirty-one base models, 4 sets of factor scores were generated for the Adult and EEG Eye State data sets and 6 sets of factor scores were generated for the Credit Card Client data set. They were combined with base models to produce the final prediction in the last ensemble step.

Table 14

Number of MCA Factor Scores

Data Set	Number of MCA Factor Scores
Adult	4
Credit Card Clients	6
EEG Eye State	4

Base Model Selection

The literature shows that choosing a subset of optimal base models based on diversity and accuracy should improve the ensemble performance (Zeng, Chao, & Wong, 2010). To improve the predictive accuracy, we applied two model selection methods in this research. One method is Cramér's V correlation analysis. The other method is backward selection based on AIC generated by logistic regression model.

Cramér's V Correlation Analysis

One model selection method is the Cramér's V correlation analysis. Cramér's V correlation coefficient was calculated for paired base models; then, paired base models with relative lower values of the correlation coefficient were selected and kept in the final ensemble step. Appendix C, D, and E list the correlation coefficients of each paired base models for the three data sets in detail. It shows that random forest base

models and extremely randomized trees base models have the most correlated prediction on both credit card clients and EEG eye state data sets, and the second correlated prediction on the adult data set. This is not surprising because these two types of models have very similar theories in producing predictions. Extreme gradient boosting model and extremely randomized trees generated the least correlated predictions on all three data sets. On the credit card client data set, extreme gradient boosting base models generated predictions which are less correlated with those generated by both random forest and extremely randomized trees base models.

It was hard to select a threshold value of correlation coefficient and finalize the number of selected base models for each data set. However, considering the average correlation between different types of base models, it was found that XGB and ERT base models had the least correlated nature on average. Thus, ten XGB and eleven ERT base models were kept as the selected base models in experiment 3 and 4. The average Cramér's V correlation coefficients of two types of base models, which are summarized from Appendix C, D, and E, are listed in table 15.

Table 15

Average Cramér's V Correlation Coefficient of Two Type of Base Models

Data Set	XGB vs. ERT	XGB vs. RF	ERT vs. RF
Adult	0.7220	0.8572	0.7829
Credit Card Clients	0.0049	0.0059	0.8313
EEG Eye State	0.8257	0.8271	0.9306

Note: XGB = Extreme Gradient Boosting; ERT = Extremely Randomized Trees; RF = Random Forest

Backward Selection Method

Another base model selection method in the research is backward selection based on AIC provided by logistic regression model. It first includes all base model outputs as variable inputs for a logistic regression ensemble. The AIC of each variable is evaluated one by one by excluding the variable in the logistic regression ensemble. The variable (base model output) that contributes the most ensemble AIC is excluded in the next round. The overall AIC of the new logistic regression ensemble was compared with that of previous logistic regression ensemble. If the decreasing of overall AIC wasn't significant at 0.05 alpha level, the backward selection procedure stopped. Table 16 summarizes the number and type of selected base models, the initial AIC with all base models, and the final AIC with only selected base models.

The backward selection procedure selected 10 base models on adult data set, 14 base models on the credit card client data set, and 15 base models on the EEG eye state data set. Extreme gradient boosting base models were selected the most on adult and EEG eye state data sets. Random forest base models were selected the most on the credit card client data set, and the least or equal least on the other two data sets. Extremely randomized trees were selected equal least on Adult and Credit Card Clients data set. Compared to the overall thirty-one base models without selection, the number of optimal subset of base models is only 50% or less in count. Overall, XGB base models were more favorite to the backward AIC selection. It makes sense because XGB base models provide better predictions in accuracy than the other two types of base models.

Table 16

Backward Selected Base Models with AIC Values

Data Set	Selected Base Models	AIC without Selection	AIC with Selection
Adult	5 XGB, 2 ERT, 2RF	11307	10634
Credit Card Clients	4 XGB, 4 ERT, 6RF	8261	8247
EEG Eye State	7 XGB, 5 ERT, 3RF	1456	1432

Experiment One: Ensemble all Base Models

In this experiment, all thirty-one base models were ensembled by four different ensemble methods, majority voting (MV), extreme gradient boosting (XGB), random forest (RF), and logistic regression (LR). The ensemble performance was compared with the benchmark and those of individual base models on the three UCI data sets. The four types of ensembles were also compared with each other.

Ensembles Compared with Benchmarks

In Table 17, the classification accuracy of the ensemble models on the three UCI test data sets are listed. Overall, all the ensemble accuracies are better than the benchmarks on all the three data sets. Random Forest ensemble method performed the best on increasing the accuracy on all the data sets. Majority voting ensemble method increased the predictive accuracy the least on all three data sets. Extreme gradient boosting and logistic regression ensemble methods had comparable performance with random forest on the Adult data set. They achieved better classification performance than majority voting method and less classification performance than random forest method on Credit Card Clients and EEG Eye State data sets. Compared with

benchmarks, majority voting ensemble method increased the classification accuracy on the three data sets from 0.05% to 1.14%; logistic regression ensemble method increased the accuracy from 0.33% to 3.54%; extreme gradient boosting ensemble method increased the accuracy from 0.61% to 3.55%; random forest ensemble method increased the accuracy from 2.36% to 4.19%.

Table 17

Ensemble Accuracy of all Base Models

Data Set	Benchmark Accuracy	Ensemble Method	Ensemble Accuracy	Accuracy Increase
Adult	0.8646	Majority Voting	0.8688	0.49%
		Random Forest	0.8957	3.60%
		Extreme Gradient Boosting	0.8953	3.55%
		Logistic Regression	0.8952	3.54%
Credit Card Clients	0.8167	Majority Voting	0.8171	0.05%
		Random Forest	0.8360	2.36%
		Extreme Gradient Boosting	0.8217	0.61%
		Logistic Regression	0.8194	0.33%
EEG Eye State	0.9282	Majority Voting	0.9388	1.14%
		Random Forest	0.9671	4.19%
		Extreme Gradient Boosting	0.9539	2.77%
		Logistic Regression	0.9522	2.59%

Ensembles Compared with Individual Base Models

Compared with individual base models, the classification accuracy provided by the majority voting ensemble is higher than those of RF and ERT base models, but lower than those of XGB base models on Adult data sets. On EEG Eye State data set, it had better performance than individual RF and XGB base models, but worse performance than ERT base models. On the Credit Card Clients data set, it had better

performance than ERT base models and half of RF base models, but worse performance than XGB base models. In summary, the majority voting ensemble method achieved better performance than only around two third individual base models. It seemed that majority voting method was not an ideal ensemble method in this experiment because the base model performance of XGB or ERT or RF outperformed its performance on different data sets. Table 18 lists the comparison in detail.

Table 18

MV Ensemble in Experiment One Compared with Base Models

Data Set	Base Model	MV Ensemble vs. RF Base Model		MV Ensemble vs. ERT Base Model		MV Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
Adult	1	0.8644	0.50%	0.8450	2.82%	0.8706	-0.21%
	2	0.8649	0.45%	0.8433	3.02%	0.8728	-0.46%
	3	0.8640	0.56%	0.8446	2.86%	0.8708	-0.23%
	4	0.8646	0.48%	0.8452	2.80%	0.8730	-0.48%
	5	0.8651	0.43%	0.8452	2.80%	0.8745	-0.65%
	6	0.8642	0.53%	0.8445	2.87%	0.8762	-0.84%
	7	0.8649	0.45%	0.8455	2.75%	0.8751	-0.72%
	8	0.8649	0.45%	0.8458	2.72%	0.8770	-0.93%
	9	0.8642	0.53%	0.8450	2.82%	0.8767	-0.90%
	10	0.8651	0.43%	0.8458	2.72%	0.8755	-0.77%
	11	N/A	N/A	0.8456	2.74%	N/A	N/A
	Average	0.8646	0.48%	0.8450	2.81%	0.8742	-0.62%
Credit Card Clients	1	0.8143	0.34%	0.8121	0.62%	0.8169	0.02%
	2	0.8170	0.01%	0.8143	0.34%	0.8203	-0.39%
	3	0.8176	-0.06%	0.8134	0.45%	0.8234	-0.77%
	4	0.8174	-0.04%	0.8130	0.50%	0.8236	-0.79%
	5	0.8172	-0.01%	0.8130	0.50%	0.8254	-1.01%
	6	0.8156	0.18%	0.8130	0.50%	0.8262	-1.10%
	7	0.8150	0.26%	0.8130	0.50%	0.8257	-1.04%
	8	0.8176	-0.06%	0.8148	0.28%	0.8258	-1.05%
	9	0.8187	-0.20%	0.8148	0.28%	0.8260	-1.08%

Data Set	Base Model	MV Ensemble vs. RF Base Model		MV Ensemble vs. ERT Base Model		MV Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
	10	0.8169	0.02%	0.8148	0.28%	0.8256	-1.03%
	11	N/A	N/A	0.8148	0.28%	N/A	N/A
	Average	0.8167	0.05%	0.8137	0.41%	0.8239	-0.82%
EEG Eye State	1	0.9243	1.57%	0.9372	0.17%	0.9009	4.21%
	2	0.9268	1.29%	0.9424	-0.38%	0.9059	3.63%
	3	0.9237	1.63%	0.9446	-0.61%	0.9003	4.28%
	4	0.9295	1.00%	0.9455	-0.71%	0.9119	2.95%
	5	0.9292	1.03%	0.9439	-0.54%	0.9105	3.11%
	6	0.9308	0.86%	0.9468	-0.84%	0.9089	3.29%
	7	0.9288	1.08%	0.9435	-0.50%	0.9061	3.61%
	8	0.9306	0.88%	0.9453	-0.69%	0.8988	4.45%
	9	0.9299	0.96%	0.9450	-0.66%	0.8925	5.19%
	10	0.9288	1.08%	0.9473	-0.90%	0.8636	8.71%
	11	N/A	N/A	0.9473	-0.90%	N/A	N/A
	Average	0.9282	1.14%	0.9444	-0.60%	0.8999	4.32%

Note: N/A = there was no data available

By comparing the ensemble performance, the extreme gradient boosting method outperformed all thirty-one base models on Adult and EEG Eye State data sets, and also outperformed majority base models excluding eight XGB base model on the Credit Card Clients data set. The XGB ensemble method increased the classification accuracy from 2.09% to 6.16% compared with the base models on the Adult data set. It also increased the classification accuracy from 0.70% to 10.46% on the EEG Eye State data set. This ensemble method increased the classification accuracy from 0.17% to 1.07% on the Credit Card Clients data set with the exception of eight XGB base models which had better performance than the ensemble method. Table 19 lists the comparison in detail.

Table 19

XGB Ensemble in Experiment One Compared with Base Models

Data Set	Base Model	XGB Ensemble vs. RF Base Model		XGB Ensemble vs. ERT Base Model		XGB Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
Adult	1	0.8644	3.57%	0.8450	5.96%	0.8706	2.83%
	2	0.8649	3.51%	0.8433	6.16%	0.8728	2.58%
	3	0.8640	3.63%	0.8446	6.00%	0.8708	2.82%
	4	0.8646	3.55%	0.8452	5.93%	0.8730	2.56%
	5	0.8651	3.49%	0.8452	5.93%	0.8745	2.38%
	6	0.8642	3.60%	0.8445	6.01%	0.8762	2.18%
	7	0.8649	3.51%	0.8455	5.89%	0.8751	2.31%
	8	0.8649	3.52%	0.8458	5.86%	0.8770	2.09%
	9	0.8642	3.60%	0.8450	5.96%	0.8767	2.12%
	10	0.8651	3.50%	0.8458	5.85%	0.8755	2.26%
	11	N/A	N/A	0.8456	5.87%	N/A	N/A
	Average	0.8646	3.55%	0.8450	5.95%	0.8742	2.41%
Credit Card Clients	1	0.8143	0.91%	0.8121	1.18%	0.8169	0.59%
	2	0.8170	0.58%	0.8143	0.91%	0.8203	0.17%
	3	0.8176	0.50%	0.8134	1.02%	0.8234	-0.21%
	4	0.8174	0.53%	0.8130	1.07%	0.8236	-0.23%
	5	0.8172	0.55%	0.8130	1.07%	0.8254	-0.45%
	6	0.8156	0.75%	0.8130	1.07%	0.8262	-0.54%
	7	0.8150	0.82%	0.8130	1.07%	0.8257	-0.48%
	8	0.8176	0.50%	0.8148	0.85%	0.8258	-0.50%
	9	0.8187	0.37%	0.8148	0.85%	0.8260	-0.52%
	10	0.8169	0.59%	0.8148	0.85%	0.8256	-0.47%
	11	N/A	N/A	0.8148	0.85%	N/A	N/A
	Average	0.8167	0.61%	0.8137	0.98%	0.8239	-0.27%
EEG Eye State	1	0.9243	3.20%	0.9372	1.78%	0.9009	5.88%
	2	0.9268	2.92%	0.9424	1.22%	0.9059	5.30%
	3	0.9237	3.27%	0.9446	0.98%	0.9003	5.95%
	4	0.9295	2.63%	0.9455	0.89%	0.9119	4.61%
	5	0.9292	2.66%	0.9439	1.06%	0.9105	4.77%
	6	0.9308	2.48%	0.9468	0.75%	0.9089	4.95%
	7	0.9288	2.70%	0.9435	1.10%	0.9061	5.28%

Data Set	Base Model	XGB Ensemble vs. RF Base Model		XGB Ensemble vs. ERT Base Model		XGB Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
	8	0.9306	2.50%	0.9453	0.91%	0.8988	6.13%
	9	0.9299	2.58%	0.9450	0.94%	0.8925	6.88%
	10	0.9288	2.70%	0.9473	0.70%	0.8636	10.46%
	11	N/A	N/A	0.9473	0.70%	N/A	N/A
	Average	0.9282	2.76%	0.9444	1.00%	0.8999	6.00%

Note: N/A = there was no data available

The ensemble performance of logistic regression method is better than all thirty-one base models on the Adult and EEG Eye State data sets, and is also better than most base models excluding nine XGB base model on the Credit Card Clients data set. The LR ensemble method increased the classification accuracy from 2.08% to 6.15% when compared with the base models on the Adult data set. It also increased the classification accuracy from 0.52% to 10.26% on the EEG Eye State data set. This ensemble method improved the classification accuracy from 0.09% to 0.9% on the Credit Card Clients data set except that nine XGB base models had better performance than the LR ensemble. The comparison in detail can be found in table 20.

Table 20

LR Ensemble in Experiment One Compared with Base Models

Data Set	Base Model	LR Ensemble vs. RF Base Model		LR Ensemble vs. ERT Base Model		LR Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
Adult	1	0.8644	3.56%	0.8450	5.94%	0.8706	2.82%
	2	0.8649	3.50%	0.8433	6.15%	0.8728	2.57%
	3	0.8640	3.62%	0.8446	5.99%	0.8708	2.81%
	4	0.8646	3.54%	0.8452	5.92%	0.8730	2.55%
	5	0.8651	3.48%	0.8452	5.92%	0.8745	2.37%

Data Set	Base Model	LR Ensemble vs. RF Base Model		LR Ensemble vs. ERT Base Model		LR Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
	6	0.8642	3.59%	0.8445	6.00%	0.8762	2.17%
	7	0.8649	3.50%	0.8455	5.88%	0.8751	2.30%
	8	0.8649	3.51%	0.8458	5.84%	0.8770	2.08%
	9	0.8642	3.59%	0.8450	5.94%	0.8767	2.11%
	10	0.8651	3.48%	0.8458	5.84%	0.8755	2.25%
	11	N/A	N/A	0.8456	5.86%	N/A	N/A
	Average	0.8646	3.54%	0.8450	5.94%	0.8742	2.40%
Credit card clients	1	0.8143	0.63%	0.8121	0.90%	0.8169	0.31%
	2	0.8170	0.29%	0.8143	0.63%	0.8203	-0.11%
	3	0.8176	0.22%	0.8134	0.74%	0.8234	-0.49%
	4	0.8174	0.24%	0.8130	0.79%	0.8236	-0.51%
	5	0.8172	0.27%	0.8130	0.79%	0.8254	-0.73%
	6	0.8156	0.47%	0.8130	0.79%	0.8262	-0.82%
	7	0.8150	0.54%	0.8130	0.79%	0.8257	-0.76%
	8	0.8176	0.22%	0.8148	0.56%	0.8258	-0.78%
	9	0.8187	0.09%	0.8148	0.56%	0.8260	-0.80%
	10	0.8169	0.31%	0.8148	0.56%	0.8256	-0.75%
	11	N/A	N/A	0.8148	0.56%	N/A	N/A
	Average	0.8167	0.33%	0.8137	0.70%	0.8239	-0.54%
EEG Eye State	1	0.9243	3.02%	0.9372	1.60%	0.9009	5.69%
	2	0.9268	2.74%	0.9424	1.04%	0.9059	5.11%
	3	0.9237	3.09%	0.9446	0.80%	0.9003	5.76%
	4	0.9295	2.44%	0.9455	0.71%	0.9119	4.42%
	5	0.9292	2.48%	0.9439	0.88%	0.9105	4.58%
	6	0.9308	2.30%	0.9468	0.57%	0.9089	4.76%
	7	0.9288	2.52%	0.9435	0.92%	0.9061	5.09%
	8	0.9306	2.32%	0.9453	0.73%	0.8988	5.94%
	9	0.9299	2.40%	0.9450	0.76%	0.8925	6.69%
	10	0.9288	2.52%	0.9473	0.52%	0.8636	10.26%
	11	N/A	N/A	0.9473	0.52%	N/A	N/A
	Average	0.9282	2.58%	0.9444	0.82%	0.8999	5.81%

Note: N/A = there was no data available

The ensemble performance of random forest method is better than all thirty-one base models on all three data sets. The RF ensemble method increased the classification accuracy from 2.14% to 6.21% when compared with the base models on the Adult data set. It improved the classification accuracy from 2.09% to 11.98% on the EEG Eye State data. This ensemble method also increased the classification accuracy from 1.19% to 2.94% on the Credit Card Clients data. The accuracies increased by RF ensemble method outperformed all the other accuracies increased by MV, XGB, and LR ensemble method. The comparison in detail is in table 21.

Table 21

RF Ensemble in Experiment One Compared with Base Models

Data Set	Base Model	RF Ensemble vs. RF Base Model		RF Ensemble vs. ERT Base Model		RF Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
Adult	1	0.8644	3.62%	0.8450	6.00%	0.8706	2.88%
	2	0.8649	3.56%	0.8433	6.21%	0.8728	2.62%
	3	0.8640	3.67%	0.8446	6.05%	0.8708	2.86%
	4	0.8646	3.59%	0.8452	5.98%	0.8730	2.60%
	5	0.8651	3.53%	0.8452	5.98%	0.8745	2.43%
	6	0.8642	3.65%	0.8445	6.06%	0.8762	2.23%
	7	0.8649	3.56%	0.8455	5.93%	0.8751	2.36%
	8	0.8649	3.56%	0.8458	5.90%	0.8770	2.14%
	9	0.8642	3.65%	0.8450	6.00%	0.8767	2.16%
	10	0.8651	3.54%	0.8458	5.90%	0.8755	2.31%
	11	N/A	N/A	0.8456	5.92%	N/A	N/A
	Average	0.8646	3.59%	0.8450	5.99%	0.8742	2.46%
Credit Card Clients	1	0.8143	2.66%	0.8121	2.94%	0.8169	2.34%
	2	0.8170	2.33%	0.8143	2.66%	0.8203	1.91%
	3	0.8176	2.25%	0.8134	2.78%	0.8234	1.53%
	4	0.8174	2.28%	0.8130	2.83%	0.8236	1.51%
	5	0.8172	2.30%	0.8130	2.83%	0.8254	1.28%
	6	0.8156	2.50%	0.8130	2.83%	0.8262	1.19%

Data Set	Base Model	RF Ensemble vs. RF Base Model		RF Ensemble vs. ERT Base Model		RF Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
	7	0.8150	2.58%	0.8130	2.83%	0.8257	1.25%
	8	0.8176	2.25%	0.8148	2.60%	0.8258	1.24%
	9	0.8187	2.11%	0.8148	2.60%	0.8260	1.21%
	10	0.8169	2.34%	0.8148	2.60%	0.8256	1.26%
	11	N/A	N/A	0.8148	2.60%	N/A	N/A
	Average	0.8167	2.36%	0.8137	2.74%	0.8239	1.47%
EEG Eye State	1	0.9243	4.63%	0.9372	3.19%	0.9009	7.35%
	2	0.9268	4.35%	0.9424	2.62%	0.9059	6.76%
	3	0.9237	4.70%	0.9446	2.38%	0.9003	7.42%
	4	0.9295	4.05%	0.9455	2.28%	0.9119	6.05%
	5	0.9292	4.08%	0.9439	2.46%	0.9105	6.22%
	6	0.9308	3.90%	0.9468	2.14%	0.9089	6.40%
	7	0.9288	4.12%	0.9435	2.50%	0.9061	6.73%
	8	0.9306	3.92%	0.9453	2.31%	0.8988	7.60%
	9	0.9299	4.00%	0.9450	2.34%	0.8925	8.36%
	10	0.9288	4.12%	0.9473	2.09%	0.8636	11.98%
	11	N/A	N/A	0.9473	2.09%	N/A	N/A
	Average	0.9282	4.19%	0.9444	2.40%	0.8999	7.46%

Note: N/A = there was no data available

Comparison of Ensemble Methods

In this experiment, majority voting ensemble showed weak ensemble power when comparing its performance with benchmarks or individual base models as reported in table 18, 19, 20, and 21. However, random forest showed very positive ability in combining all base models. It outperformed all the benchmarks and base models on all three data sets. Extreme gradient boosting and logistic regression had a comparable performance in ensemble. They outperformed the benchmarks and majority of base models except for several XGB base models which had better

performance. They didn't perform well when compared with RF ensemble, but did have better performance than MV ensemble. In table 22, RF, XGB, and LR ensemble are compared with MV ensemble. The increased accuracy in percentage is reported. MV ensemble was chosen to be compared since it had the least ensemble accuracy. We would like to see how much the other three ensembles are better than it. It shows that RF ensemble has 3.10%, 2.31%, and 3.01% better performance than MV on the Adult, Credit Card Clients, and EEG Eye State data set. XGB ensemble has 3.05%, 0.56%, and 1.61% better performance than MV ensemble on those three data sets. LR ensemble has 3.04%, 0.28%, and 1.43% better performance than MV ensemble on those three data sets. In summary, random forest ensemble is the best method of combining all base models.

Table 22

Ensemble Comparison in Experiment One

Data Set	Ensemble Method	Ensemble Accuracy	Accuracy Comparison with MV Ensemble
Adult	Majority Voting	0.8688	N/A
	Random Forest	0.8957	3.10%
	Extreme Gradient Boosting	0.8953	3.05%
	Logistic Regression	0.8952	3.04%
Credit Card Clients	Majority Voting	0.8171	N/A
	Random Forest	0.8360	2.31%
	Extreme Gradient Boosting	0.8217	0.56%
	Logistic Regression	0.8194	0.28%
EEG Eye State	Majority Voting	0.9388	N/A
	Random Forest	0.9671	3.01%
	Extreme Gradient Boosting	0.9539	1.61%
	Logistic Regression	0.9522	1.43%

Note: N/A = there was no data available

Experiment Two: Ensemble all Base Models and MCA Factor Scores

In addition to combining all thirty-one base models as in experiment one, factor scores of multiple correspondence analysis were added and integrated with the thirty-one base models by three different ensemble methods, extreme gradient boosting, random forest, and logistic regression. Majority voting ensemble is not applicable as an ensemble method here because factor scores are numerical variables and not presented as prediction of target variable.

Ensembles Compared with Individual Base Model

Comparing the ensemble performance of the logistic regression method with those of individual base models, it was noticed that the ensemble performance of logistic regression didn't change whether the factors scores of multiple correspondence analysis were added or not. Its performance was the same as that in experiment design one reported in Table 20. It outperformed all thirty-one base models on the Adult and EEG Eye State data sets, and also outperformed majority base models except for nine extreme gradient boosting base model on Credit Card Clients data set.

The ensemble performance of extreme gradient boosting method is better than all thirty-one base models on the Adult and EEG Eye State data sets, and outperforms most base models except for eight XGB base models on the Credit Card Clients data set. The XGB ensemble increased classification accuracies from 2.09% to 6.16% when compared with the base models on the Adult data set. It improved the classification accuracy from 0.75% to 10.51% on the EEG Eye State data set. This ensemble method also increased the classification accuracy from 0.26% to 1.16% on the Credit Card

Clients data set except that eight extreme gradient boosting base models had better performance than the ensemble method. Table 23 lists the comparison in detail.

Table 23

XGB Ensemble in Experiment Two Compared with Base Models

Data Set	Base Model	XGB Ensemble vs. RF Base Model		XGB Ensemble vs. ERT Base Model		XGB Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
Adult	1	0.8644	3.57%	0.8450	5.96%	0.8706	2.83%
	2	0.8649	3.51%	0.8433	6.16%	0.8728	2.58%
	3	0.8640	3.63%	0.8446	6.00%	0.8708	2.82%
	4	0.8646	3.55%	0.8452	5.93%	0.8730	2.56%
	5	0.8651	3.49%	0.8452	5.93%	0.8745	2.38%
	6	0.8642	3.60%	0.8445	6.01%	0.8762	2.18%
	7	0.8649	3.51%	0.8455	5.89%	0.8751	2.31%
	8	0.8649	3.52%	0.8458	5.86%	0.8770	2.09%
	9	0.8642	3.60%	0.8450	5.96%	0.8767	2.12%
	10	0.8651	3.50%	0.8458	5.85%	0.8755	2.26%
	11	N/A	N/A	0.8456	5.87%	N/A	N/A
	Average	0.8646	3.55%	0.8450	5.95%	0.8742	2.41%
Credit Card Clients	1	0.8143	0.99%	0.8121	1.27%	0.8169	0.67%
	2	0.8170	0.66%	0.8143	0.99%	0.8203	0.26%
	3	0.8176	0.59%	0.8134	1.11%	0.8234	-0.12%
	4	0.8174	0.61%	0.8130	1.16%	0.8236	-0.15%
	5	0.8172	0.64%	0.8130	1.16%	0.8254	-0.36%
	6	0.8156	0.83%	0.8130	1.16%	0.8262	-0.46%
	7	0.8150	0.91%	0.8130	1.16%	0.8257	-0.40%
	8	0.8176	0.59%	0.8148	0.93%	0.8258	-0.41%
	9	0.8187	0.45%	0.8148	0.93%	0.8260	-0.44%
	10	0.8169	0.67%	0.8148	0.93%	0.8256	-0.39%
	11	N/A	N/A	0.8148	0.93%	N/A	N/A
	Average	0.8167	0.69%	0.8137	1.07%	0.8239	-0.18%
EEG Eye State	1	0.9243	3.26%	0.9372	1.84%	0.9009	5.94%
	2	0.9268	2.98%	0.9424	1.27%	0.9059	5.35%
	3	0.9237	3.32%	0.9446	1.04%	0.9003	6.01%
	4	0.9295	2.68%	0.9455	0.94%	0.9119	4.66%

Data Set	Base Model	XGB Ensemble vs. RF Base Model		XGB Ensemble vs. ERT Base Model		XGB Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
	5	0.9292	2.71%	0.9439	1.11%	0.9105	4.82%
	6	0.9308	2.54%	0.9468	0.80%	0.9089	5.01%
	7	0.9288	2.76%	0.9435	1.16%	0.9061	5.33%
	8	0.9306	2.56%	0.9453	0.96%	0.8988	6.19%
	9	0.9299	2.63%	0.9450	0.99%	0.8925	6.94%
	10	0.9288	2.76%	0.9473	0.75%	0.8636	10.51%
	11	N/A	N/A	0.9473	0.75%	N/A	N/A
	Average	0.9282	2.82%	0.9444	1.05%	0.8999	6.05%

Note: N/A = there was no data available

The ensemble performance by random forest method outperforms all thirty-one base models on all of the three data sets. After integrating factor scores of multiple correspondence analysis with base models, the performance of random forest ensemble method continued to be the one that provided the best classification accuracy. The RF ensemble method increased the classification accuracy from 2.58% to 6.67% when compared with base models on the Adult data set. It also increased the classification accuracy from 2.74% to 12.70% on the EEG Eye State data. This ensemble method increased the classification accuracy from 1.65% to 3.41% on the Credit Card Clients data. Table 24 summarizes the comparison in detail.

Table 24

RF Ensemble in Experiment Two Compared with Base Models

Data Set	Base Model	RF Ensemble vs. RF Base Model		RF Ensemble vs. Extremely ERT Base Model		RF Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
Adult	1	0.8644	4.07%	0.8450	6.46%	0.8706	3.33%
	2	0.8649	4.01%	0.8433	6.67%	0.8728	3.07%
	3	0.8640	4.13%	0.8446	6.51%	0.8708	3.31%
	4	0.8646	4.04%	0.8452	6.44%	0.8730	3.05%
	5	0.8651	3.99%	0.8452	6.44%	0.8745	2.88%
	6	0.8642	4.10%	0.8445	6.52%	0.8762	2.67%
	7	0.8649	4.01%	0.8455	6.40%	0.8751	2.80%
	8	0.8649	4.02%	0.8458	6.36%	0.8770	2.58%
	9	0.8642	4.10%	0.8450	6.46%	0.8767	2.61%
	10	0.8651	3.99%	0.8458	6.36%	0.8755	2.75%
	11	N/A	N/A	0.8456	6.38%	N/A	N/A
	Average	0.8646	4.04%	0.8450	6.46%	0.8742	2.90%
Credit card clients	1	0.8143	3.13%	0.8121	3.41%	0.8169	2.80%
	2	0.8170	2.79%	0.8143	3.13%	0.8203	2.38%
	3	0.8176	2.72%	0.8134	3.25%	0.8234	1.99%
	4	0.8174	2.74%	0.8130	3.30%	0.8236	1.97%
	5	0.8172	2.77%	0.8130	3.30%	0.8254	1.74%
	6	0.8156	2.97%	0.8130	3.30%	0.8262	1.65%
	7	0.8150	3.04%	0.8130	3.30%	0.8257	1.71%
	8	0.8176	2.72%	0.8148	3.07%	0.8258	1.70%
	9	0.8187	2.58%	0.8148	3.07%	0.8260	1.67%
	10	0.8169	2.80%	0.8148	3.07%	0.8256	1.72%
	11	N/A	N/A	0.8148	3.07%	N/A	N/A
	Average	0.8167	2.82%	0.8137	3.20%	0.8239	1.93%
EEG Eye State	1	0.9243	5.30%	0.9372	3.85%	0.9009	8.04%
	2	0.9268	5.02%	0.9424	3.28%	0.9059	7.44%
	3	0.9237	5.37%	0.9446	3.04%	0.9003	8.11%
	4	0.9295	4.71%	0.9455	2.94%	0.9119	6.73%
	5	0.9292	4.75%	0.9439	3.11%	0.9105	6.90%
	6	0.9308	4.57%	0.9468	2.80%	0.9089	7.09%
	7	0.9288	4.79%	0.9435	3.16%	0.9061	7.42%

Data Set	Base Model	RF Ensemble vs. RF Base Model		RF Ensemble vs. Extremely ERT Base Model		RF Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
	8	0.9306	4.59%	0.9453	2.96%	0.8988	8.29%
	9	0.9299	4.67%	0.9450	2.99%	0.8925	9.05%
	10	0.9288	4.79%	0.9473	2.74%	0.8636	12.70%
	11	N/A	N/A	0.9473	2.74%	N/A	N/A
	Average	0.9282	4.85%	0.9444	3.06%	0.8999	8.15%

Note: N/A = there was no data available

Ensembles Compared with Benchmarks and Experiment One

In this experiment, MCA factors scores are involved in the ensemble approaches. Adding MCA factor scores to XGB ensemble increased the ensemble performance on the Credit Card Client and EEG Eye State data sets, and kept almost the same accuracy on the Adult data set; it did improve the RF ensemble performance; however, it didn't impact the LR ensemble performance at all on any of the data sets.

Table 25 summarizes the classification accuracy of the ensemble models and their comparison with those of benchmarks and experiment one on the three UCI test data sets. Overall, all the ensemble methods in experiment two outperform the benchmarks on all three data sets. They also outperform or have the same performance as the same ensemble methods in experiment one. Comparing with benchmarks, LR ensemble method increased the accuracy by 0.33%, 2.59% and 3.54% on the three data sets; XGB ensemble method increased the accuracy by 0.70%, 2.82% and 3.55%; RF ensemble method increased the accuracy by 2.83%, 4.05% and 4.86%.

Comparing with the same ensemble methods in experiment one, LR ensemble method had the same classification accuracies on all the three data sets; XGB ensemble

method kept the same accuracy on the Adult data set, and increased the accuracy by 0.05% and 0.09% on the other two data sets; RF ensemble method increased the accuracy by 0.44%, 0.45% and 0.64% on the three data sets. Here, we conclude that factor scores of MCA help increase classification accuracy of RF ensemble method; they might also help increase the performance of XGB ensemble method; however, they have no impact on the performance of LR ensemble method.

Table 25

Experiment Two Compared to Benchmarks and Experiment One

Data Set	Ensemble Method	Ensemble Accuracy		Accuracy Increase	
		Exp 1	Exp2	Exp 2 vs. Benchmark	Exp2 vs. Exp1
Adult	RF	0.8957	0.8996	4.05%	0.44%
	XGB	0.8953	0.8953	3.55%	0.00%
	LR	0.8952	0.8952	3.54%	0.00%
Credit Card Clients	RF	0.8360	0.8398	2.83%	0.45%
	XGB	0.8217	0.8224	0.70%	0.09%
	LR	0.8194	0.8194	0.33%	0.00%
EEG Eye State	RF	0.9671	0.9733	4.86%	0.64%
	XGB	0.9539	0.9544	2.82%	0.05%
	LR	0.9522	0.9522	2.59%	0.00%

Note: Exp 1 means experiment one; Exp 2 means experiment two.

Comparison of Ensemble Methods

Combining MCA factor scores with all base models, LR ensemble performed the same as in experiment one. Its performance is comparable to but a little bit worse than XGB ensemble and much worse than RF ensemble method when compared with benchmarks or experiment one. LR and XGB outperformed the benchmarks and majority of base models except that several XGB base models had better performance

than them. RF ensemble performed the best among the three ensembles. It continued to be the most powerful ensemble method in experiment two. The increased accuracies by RF ensemble method outperformed all the other increased accuracies by XGB and LR ensemble method on all three data sets as shown in table 23, 24, and 25.

Table 26

Ensemble Comparison in Experiment Two

Data Set	Ensemble Method	Ensemble Accuracy	Accuracy Increased from LR Ensemble
Adult	Logistic Regression	0.8952	N/A
	Random Forest	0.8996	0.49%
	Extreme Gradient Boosting	0.8953	0.01%
Credit Card Clients	Logistic Regression	0.8194	N/A
	Random Forest	0.8398	2.49%
	Extreme Gradient Boosting	0.8224	0.37%
EEG Eye State	Logistic Regression	0.9522	N/A
	Random Forest	0.9733	2.22%
	Extreme Gradient Boosting	0.9544	0.23%

Note: N/A = there was no data available

RF and XGB ensembles are compared with LR ensemble in table 26. The improved accuracy in percentage is reported. LR ensemble was chosen to be the baseline of the comparison since it held the least ensemble accuracy in this experiment. How much better the RF and XGB ensembles are than the LR ensemble is shown in table 26. It shows that RF ensemble has 0.49%, 2.49%, and 2.22% better performance than LR on the Adult, Credit Card Clients, and EEG Eye State data set. XGB ensemble has 0.01%, 0.37%, and 0.23% better performance than LR ensemble on the three data sets. In summary, random forest ensemble is the best method of combining all base models and MCA factor scores.

Experiment Three: Ensemble all Base Models with Model Selections

Without considering the effect of MCA factor scores, this experiment ensembled only optimal subset of base models selected by Cramér's V correlation analysis and backward AIC selection. Majority voting, extreme gradient boosting, and random forest method worked as ensemble methods for twenty-one base models selected by Cramér's V correlation analysis. These twenty-one selected base models are XGB and ERT base models. These two types of base models have relatively less correlation on average. Logistic regression works as an ensemble method for optimal selected base models which are chosen by backward selection method based on the AIC value of the logistic regression model. The backward selection procedure first combines all thirty-one base models, selects one base model that contributes the most AIC of logistic regression, and then removes it in the next round of selection. The backward selection stops when the overall AIC of logistic regression doesn't significantly decrease at a 0.05 alpha level. The number and the types of selected base models of different data sets are not fixed but determined by the backward selection procedure and the nature of data sets.

Ensembles Compared with Individual Base Models

Compared with individual base models, the classification accuracy provided by the MV ensemble method is higher than those of ERT base models, but is lower than those of RF and XGB base models on the Adult data set. For the EEG Eye State data set, it has better performance than individual RF, XGB base models, and five ERT base models. On the Credit Card Clients data set, it has better performance than ERT base

models and one RF base models, but worse performance than XGB and nine RF base models. In summary, the MV ensemble method only achieved better performance than half of individual base models. Most RF and XGB base models have better performance than the MV ensemble. The performance decreasing ranges from 0.07% to 2.80%. It seems that majority voting with Cramér's V model selection is not a good ensemble method in this experiment. Table 27 summarizes the comparison in detail.

Table 27

MV Ensemble in Experiment Three Compared with Base Models

Data Set	Base Model	MV Ensemble vs. RF Base Model		MV Ensemble vs. ERT Base Model		MV Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
Adult	1	0.8644	-1.39%	0.8450	0.88%	0.8706	-2.10%
	2	0.8649	-1.45%	0.8433	1.08%	0.8728	-2.34%
	3	0.8640	-1.34%	0.8446	0.92%	0.8708	-2.11%
	4	0.8646	-1.41%	0.8452	0.86%	0.8730	-2.36%
	5	0.8651	-1.47%	0.8452	0.86%	0.8745	-2.52%
	6	0.8642	-1.37%	0.8445	0.93%	0.8762	-2.71%
	7	0.8649	-1.45%	0.8455	0.81%	0.8751	-2.59%
	8	0.8649	-1.44%	0.8458	0.78%	0.8770	-2.80%
	9	0.8642	-1.37%	0.8450	0.88%	0.8767	-2.77%
	10	0.8651	-1.46%	0.8458	0.78%	0.8755	-2.64%
	11	N/A	N/A	0.8456	0.80%	N/A	N/A
	Average	0.8646	-1.41%	0.8450	0.87%	0.8742	-2.49%
Credit Card Clients	1	0.8143	0.09%	0.8121	0.36%	0.8169	-0.23%
	2	0.8170	-0.24%	0.8143	0.09%	0.8203	-0.65%
	3	0.8176	-0.32%	0.8134	0.20%	0.8234	-1.02%
	4	0.8174	-0.29%	0.8130	0.25%	0.8236	-1.04%
	5	0.8172	-0.27%	0.8130	0.25%	0.8254	-1.26%
	6	0.8156	-0.07%	0.8130	0.25%	0.8262	-1.36%
	7	0.8150	0.00%	0.8130	0.25%	0.8257	-1.30%
	8	0.8176	-0.32%	0.8148	0.02%	0.8258	-1.31%
	9	0.8187	-0.45%	0.8148	0.02%	0.8260	-1.33%

Data Set	Base Model	MV Ensemble vs. RF Base Model		MV Ensemble vs. ERT Base Model		MV Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
	10	0.8169	-0.23%	0.8148	0.02%	0.8256	-1.28%
	11	N/A	N/A	0.8148	0.02%	N/A	N/A
	Average	0.8167	-0.21%	0.8137	0.16%	0.8239	-1.08%
EEG Eye State	1	0.9243	2.20%	0.9372	0.79%	0.9009	4.85%
	2	0.9268	1.92%	0.9424	0.23%	0.9059	4.27%
	3	0.9237	2.26%	0.9446	0.00%	0.9003	4.92%
	4	0.9295	1.62%	0.9455	-0.10%	0.9119	3.59%
	5	0.9292	1.66%	0.9439	0.07%	0.9105	3.75%
	6	0.9308	1.48%	0.9468	-0.23%	0.9089	3.93%
	7	0.9288	1.70%	0.9435	0.12%	0.9061	4.25%
	8	0.9306	1.50%	0.9453	-0.07%	0.8988	5.10%
	9	0.9299	1.58%	0.9450	-0.04%	0.8925	5.84%
	10	0.9288	1.70%	0.9473	-0.29%	0.8636	9.38%
	11	N/A	N/A	0.9473	-0.29%	N/A	N/A
	Average	0.9282	1.76%	0.9444	0.02%	0.8999	4.96%

Note: N/A = there was no data available

The ensemble performance of XGB method is better than all thirty-one base models on the EEG Eye State data set, twenty-nine base models on the Adult data set, and three base models on the Credit Card Clients data set. Table 28 shows that twenty base models in total on the three data sets provided higher classification accuracy than the ensemble method in this experiment. The XGB ensemble method increased the classification accuracy by 0.54% to 10.28% from base models on the EEG Eye State data set. It worked pretty well on this data set when combining only the selected twenty-one base models. The XGB ensemble method has the increased classification accuracies by 0.07% to 3.89% on the Adult data set except for two XGB base models which have better performance than the ensemble method. However, on the Credit Card

Clients data set, seventeen RF or XGB base models have better performance than the ensemble method. It seems that XGB ensemble with Cramér's V base model selection is not an ideal ensemble method because it might perform well on some types of data, but not on other types of data. Table 28 lists the comparison in detail.

Table 28

XGB Ensemble in Experiment Three Compared with Base Models

Data Set	Base Model	XGB Ensemble vs. RF Base Model		XGB Ensemble vs. ERT Base Model		XGB Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
Adult	1	0.8644	1.35%	0.8450	3.68%	0.8706	0.63%
	2	0.8649	1.29%	0.8433	3.89%	0.8728	0.38%
	3	0.8640	1.41%	0.8446	3.73%	0.8708	0.61%
	4	0.8646	1.33%	0.8452	3.66%	0.8730	0.36%
	5	0.8651	1.27%	0.8452	3.66%	0.8745	0.19%
	6	0.8642	1.38%	0.8445	3.74%	0.8762	-0.01%
	7	0.8649	1.29%	0.8455	3.62%	0.8751	0.12%
	8	0.8649	1.30%	0.8458	3.59%	0.8770	-0.10%
	9	0.8642	1.38%	0.8450	3.68%	0.8767	-0.07%
	10	0.8651	1.28%	0.8458	3.58%	0.8755	0.07%
	11	N/A	N/A	0.8456	3.60%	N/A	N/A
	Average	0.8646	1.33%	0.8450	3.67%	0.8742	0.22%
Credit Card Clients	1	0.8143	0.28%	0.8121	0.55%	0.8169	-0.04%
	2	0.8170	-0.05%	0.8143	0.28%	0.8203	-0.45%
	3	0.8176	-0.12%	0.8134	0.39%	0.8234	-0.83%
	4	0.8174	-0.10%	0.8130	0.44%	0.8236	-0.85%
	5	0.8172	-0.07%	0.8130	0.44%	0.8254	-1.07%
	6	0.8156	0.12%	0.8130	0.44%	0.8262	-1.16%
	7	0.8150	0.20%	0.8130	0.44%	0.8257	-1.10%
	8	0.8176	-0.12%	0.8148	0.22%	0.8258	-1.11%
	9	0.8187	-0.26%	0.8148	0.22%	0.8260	-1.14%
	10	0.8169	-0.04%	0.8148	0.22%	0.8256	-1.09%
	11	N/A	N/A	0.8148	0.22%	N/A	N/A
	Average	0.8167	-0.02%	0.8137	0.35%	0.8239	-0.88%

Data Set	Base Model	XGB Ensemble vs. RF Base Model		XGB Ensemble vs. ERT Base Model		XGB Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
EEG Eye State	1	0.9243	3.04%	0.9372	1.62%	0.9009	5.72%
	2	0.9268	2.76%	0.9424	1.06%	0.9059	5.13%
	3	0.9237	3.11%	0.9446	0.83%	0.9003	5.79%
	4	0.9295	2.46%	0.9455	0.73%	0.9119	4.44%
	5	0.9292	2.50%	0.9439	0.90%	0.9105	4.60%
	6	0.9308	2.32%	0.9468	0.59%	0.9089	4.79%
	7	0.9288	2.54%	0.9435	0.94%	0.9061	5.11%
	8	0.9306	2.34%	0.9453	0.75%	0.8988	5.96%
	9	0.9299	2.42%	0.9450	0.78%	0.8925	6.71%
	10	0.9288	2.54%	0.9473	0.54%	0.8636	10.28%
	11	N/A	N/A	0.9473	0.54%	N/A	N/A
	Average	0.9282	2.60%	0.9444	0.84%	0.8999	5.83%

Note: N/A = there was no data available

The ensemble performance of random forest method outperforms all thirty-one base models on the Adult and EEG Eye State data sets. The RF ensemble method increased the classification accuracy from 1.29% to 5.33% when compared with the base models on the Adult data set. It also increased the classification accuracy from 1.52% to 11.36% on the EEG Eye State data. The RF ensemble method increased the classification accuracy from 0.06% to 0.74% on the Credit Card Clients data set except for ten base models, nine XGBs and one RF. Table 29 lists the comparisons in detail. Note that, in experiment one and two, RF ensemble outperforms all base models on all three test data sets. Here, we can only conclude that Cramér's V base model selection didn't help in increasing ensemble accuracy of RF, XGB, and MV ensemble methods.

Table 29

RF Ensemble in Experiment Three Compared with Base Models

Data Set	Base Model	RF Ensemble vs. RF Base Model		RF Ensemble vs. ERT Base Model		RF Ensemble vs XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
Adult	1	0.8644	2.76%	0.8450	5.13%	0.8706	2.03%
	2	0.8649	2.70%	0.8433	5.33%	0.8728	1.78%
	3	0.8640	2.82%	0.8446	5.17%	0.8708	2.01%
	4	0.8646	2.74%	0.8452	5.10%	0.8730	1.75%
	5	0.8651	2.68%	0.8452	5.10%	0.8745	1.58%
	6	0.8642	2.79%	0.8445	5.18%	0.8762	1.38%
	7	0.8649	2.70%	0.8455	5.06%	0.8751	1.51%
	8	0.8649	2.71%	0.8458	5.03%	0.8770	1.29%
	9	0.8642	2.79%	0.8450	5.13%	0.8767	1.32%
	10	0.8651	2.69%	0.8458	5.02%	0.8755	1.46%
	11	N/A	N/A	0.8456	5.04%	N/A	N/A
	Average	0.8646	2.74%	0.8450	5.12%	0.8742	1.61%
Credit Card Clients	1	0.8143	0.47%	0.8121	0.74%	0.8169	0.15%
	2	0.8170	0.13%	0.8143	0.47%	0.8203	-0.27%
	3	0.8176	0.06%	0.8134	0.58%	0.8234	-0.64%
	4	0.8174	0.09%	0.8130	0.63%	0.8236	-0.67%
	5	0.8172	0.11%	0.8130	0.63%	0.8254	-0.88%
	6	0.8156	0.31%	0.8130	0.63%	0.8262	-0.98%
	7	0.8150	0.38%	0.8130	0.63%	0.8257	-0.92%
	8	0.8176	0.06%	0.8148	0.41%	0.8258	-0.93%
	9	0.8187	-0.07%	0.8148	0.41%	0.8260	-0.96%
	10	0.8169	0.15%	0.8148	0.41%	0.8256	-0.91%
	11	N/A	N/A	0.8148	0.41%	N/A	N/A
	Average	0.8167	0.17%	0.8137	0.54%	0.8239	-0.70%
EEG Eye State	1	0.9243	4.05%	0.9372	2.61%	0.9009	6.75%
	2	0.9268	3.77%	0.9424	2.05%	0.9059	6.16%
	3	0.9237	4.11%	0.9446	1.81%	0.9003	6.82%
	4	0.9295	3.46%	0.9455	1.71%	0.9119	5.46%
	5	0.9292	3.50%	0.9439	1.89%	0.9105	5.62%
	6	0.9308	3.32%	0.9468	1.57%	0.9089	5.81%
	7	0.9288	3.54%	0.9435	1.93%	0.9061	6.14%

Data Set	Base Model	RF Ensemble vs. RF Base Model		RF Ensemble vs. ERT Base Model		RF Ensemble vs XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
	8	0.9306	3.34%	0.9453	1.73%	0.8988	7.00%
	9	0.9299	3.42%	0.9450	1.77%	0.8925	7.75%
	10	0.9288	3.54%	0.9473	1.52%	0.8636	11.36%
	11	N/A	N/A	0.9473	1.52%	N/A	N/A
	Average	0.9282	3.60%	0.9444	1.83%	0.8999	6.86%

Note: N/A = there was no data available

Logistic regression is the only ensemble method that integrated with backward selection in this experiment. Its performance is comparable to logistic regression that ensembles all base models in previous experiments. It outperforms all thirty-one base models on the Adult and EEG Eye State data sets, and outperforms majority base models except for nine extreme gradient boosting base models on the Credit Card Clients data set. The LR ensemble method increased the classification accuracy from 2.08% to 6.15% when compared with the base models on the Adult data set. It also increased the classification accuracy from 0.56% to 10.31% on the EEG Eye State data set. The LR ensemble method increased the classification accuracy from 0.05% to 0.86% on the Credit Card Clients data set except for nine XGB base models. It was noticed that the ensemble performance of logistic regression method kept the same level in increasing the classification accuracy from base models although it ensembled only the optimal subset of base model that had less than 50% of base models. Its performance is listed in Table 30 and turns out to be the best ensemble method in experiment three.

Table 30

LR Ensemble in Experiment Three Compared with Base Models

Data Set	Base Model	LR Ensemble vs. RF Base Model		LR Ensemble vs. ERT Base Model		LR Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
Adult	1	0.8644	3.56%	0.8450	5.94%	0.8706	2.82%
	2	0.8649	3.50%	0.8433	6.15%	0.8728	2.57%
	3	0.8640	3.62%	0.8446	5.99%	0.8708	2.81%
	4	0.8646	3.54%	0.8452	5.92%	0.8730	2.55%
	5	0.8651	3.48%	0.8452	5.92%	0.8745	2.37%
	6	0.8642	3.59%	0.8445	6.00%	0.8762	2.17%
	7	0.8649	3.50%	0.8455	5.88%	0.8751	2.30%
	8	0.8649	3.51%	0.8458	5.84%	0.8770	2.08%
	9	0.8642	3.59%	0.8450	5.94%	0.8767	2.11%
	10	0.8651	3.48%	0.8458	5.84%	0.8755	2.25%
	11	N/A	N/A	0.8456	5.86%	N/A	N/A
	Average	0.8646	3.54%	0.8450	5.94%	0.8742	2.40%
Credit Card Clients	1	0.8143	0.59%	0.8121	0.86%	0.8169	0.27%
	2	0.8170	0.26%	0.8143	0.59%	0.8203	-0.15%
	3	0.8176	0.18%	0.8134	0.70%	0.8234	-0.52%
	4	0.8174	0.21%	0.8130	0.75%	0.8236	-0.55%
	5	0.8172	0.23%	0.8130	0.75%	0.8254	-0.76%
	6	0.8156	0.43%	0.8130	0.75%	0.8262	-0.86%
	7	0.8150	0.50%	0.8130	0.75%	0.8257	-0.80%
	8	0.8176	0.18%	0.8148	0.53%	0.8258	-0.81%
	9	0.8187	0.05%	0.8148	0.53%	0.8260	-0.84%
	10	0.8169	0.27%	0.8148	0.53%	0.8256	-0.79%
	11	N/A	N/A	0.8148	0.53%	N/A	N/A
	Average	0.8167	0.29%	0.8137	0.66%	0.8239	-0.58%
EEG Eye State	1	0.9243	3.06%	0.9372	1.64%	0.9009	5.74%
	2	0.9268	2.78%	0.9424	1.08%	0.9059	5.16%
	3	0.9237	3.13%	0.9446	0.85%	0.9003	5.81%
	4	0.9295	2.49%	0.9455	0.75%	0.9119	4.46%
	5	0.9292	2.52%	0.9439	0.92%	0.9105	4.62%
	6	0.9308	2.34%	0.9468	0.61%	0.9089	4.81%
	7	0.9288	2.56%	0.9435	0.96%	0.9061	5.13%

Data Set	Base Model	LR Ensemble vs. RF Base Model		LR Ensemble vs. ERT Base Model		LR Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
	8	0.9306	2.36%	0.9453	0.77%	0.8988	5.99%
	9	0.9299	2.44%	0.9450	0.80%	0.8925	6.73%
	10	0.9288	2.56%	0.9473	0.56%	0.8636	10.31%
	11	N/A	N/A	0.9473	0.56%	N/A	N/A
	Average	0.9282	2.62%	0.9444	0.86%	0.8999	5.85%

Note: N/A = there was no data available

Ensembles Compared with Benchmarks, Experiment One and Two

Table 31 lists the classification accuracy of the ensemble models and comparison with benchmarks, experiment one, and experiment two on the three UCI test data sets. Compared with the benchmarks, unlike experiment one and two, the ensemble methods in this experiment don't outperform all of the benchmarks on the three data sets. MV ensemble only outperformed benchmark on the EEG Eye State data set and is worse on the other two data sets. XGB ensemble outperforms benchmarks on the Adult and EEG Eye State data sets but is worse on the third data set. RF and LR ensemble still outperforms benchmarks on all three data sets.

Compared with the same ensemble methods in experiment one, LR ensemble method has the same or comparable accuracies on all three data sets. It achieved a 0.04% increased accuracy on the EEG Eye State data set, and a 0.04% decreased accuracy on the Credit Card Client data set; it has the same accuracy as that in experiment one. XGB ensemble method has all decreased -2.14%, -0.62%, and -0.16% accuracy on the Adult, Credit Card Client and EEG Eye Status data sets respectively. RF ensemble method decreased the accuracy by -0.83%, -2.14%, and -0.56% on the

three data sets. MV ensemble method achieved 0.62% increased accuracy on the EEG Eye Status data set and -1.89% and -0.26% decreased accuracy on the Adult and Credit Card Client data sets. Overall, compared with the same ensemble method in the first experiment in which all base models were ensembled, nine ensemble methods with model selection on the three data sets were defeated in accuracy. Only the MV ensemble with twenty-one selected base models and LR ensemble with fifteen selected base models on the the EEG Eye State data set outperformed the same ensemble method in the first experiment.

Compared with the same ensemble methods in the second experiment, LR ensemble method has the same or comparable accuracies on all three data sets. It achieved a 0.04% increased accuracy on the EEG Eye State data set, and a 0.04% decreased accuracy on the Credit Card Client data set, so it has the same accuracy as that in the second experiment. The LR ensemble performed very stably. XGB ensemble method has all decreased -2.14%, -0.71%, and -0.21% accuracy on the Adult, Credit Card Client, and EEG Eye Status data sets. RF ensemble method decreased the accuracy by -1.26%, -2.58%, and -1.19% on the three data sets. In summary, compared with the same ensemble method in the second experiment in which all base models and MCA factor scores are ensembled, seven ensemble methods with model selection on the three data sets were defeated in accuracy. The LR ensemble with nine selected base models on the Adult data set achieved the same accuracy as the same method in experiment two. Also, the LR ensemble with fifteen selected base models on the EEG Eye State data set outperforms the same ensemble method in the second experiment.

It shows that ensemble methods in experiment two with all thirty-one base models and MCA factor scores combined have better classification accuracy than the ensemble methods in experiment one with only all the thirty-one base models combined and those in experiment three in which only selected base models are combined. Among the ensemble methods, RF ensemble models have better classification performance than majority voting, extreme gradient boosting, and logistic regression model in experiment one, two and three. LR ensemble produced very stable performance in experiment one, two, and three. Especially, LR ensemble with base model selection although integrated less than 50% of base models, but achieved about the same accuracy as it integrated all base models in experiment one and two.

Table 31

Experiment Three Compared to Benchmarks, Experiment One and Two

Data Set	Ensemble Method	Ensemble Accuracy			Accuracy Increase		
		Exp 1	Exp2	Exp3	Exp 3 vs. BMK	Exp3 vs. Exp1	Exp3 vs. Exp2
Adult	MV	0.8688	N/A	0.8524	-1.41%	-1.89%	N/A
	RF	0.8957	0.8996	0.8883	2.74%	-0.83%	-1.26%
	XGB	0.8953	0.8953	0.8761	1.33%	-2.14%	-2.14%
	LR	0.8952	0.8952	0.8952	3.54%	0.00%	0.00%
Credit Card Clients	MV	0.8171	N/A	0.8150	-0.21%	-0.26%	N/A
	RF	0.8360	0.8398	0.8181	0.17%	-2.14%	-2.58%
	XGB	0.8217	0.8224	0.8166	-0.01%	-0.62%	-0.71%
	LR	0.8194	0.8194	0.8191	0.29%	-0.04%	-0.04%
EEG Eye State	MV	0.9388	N/A	0.9446	1.77%	0.62%	N/A
	RF	0.9671	0.9733	0.9617	3.61%	-0.56%	-1.19%
	XGB	0.9539	0.9544	0.9524	2.61%	-0.16%	-0.21%
	LR	0.9522	0.9522	0.9526	2.63%	0.04%	0.04%

Note: BMK = Benchmark; Exp = Experiment Design; MV = Majority Voting; RF = Random Forest; XGB = Extreme Gradient Boosting; LR =Logistic Regression; N/A= there was no data available

Comparison of Ensemble Methods

Combining only selected base models, RF ensemble method with Cramér's V model selection didn't perform the best on all the data sets like experiment one and two. It only provided the best accuracy on the EEG Eye State data set. It is outperformed by LR ensemble with backward selected base models on the Adult and Credit Card Client data sets. LR ensemble has the same predictive power as in experiment one and two, while RF ensemble has decreased predictive ability when compared with themselves in experiment one and two. MV ensemble performs the worst on all three data sets. XGB ensemble has worse classification performance than RF and LR method, but performs better than MV ensemble method.

Table 32

Ensemble Comparison in Experiment Three

Data Set	Ensemble Method	Ensemble Accuracy	Accuracy Increased from MV Ensemble
Adult	MV	0.8524	N/A
	RF	0.8883	4.21%
	XGB	0.8761	2.78%
	LR	0.8952	5.02%
Credit Card Clients	MV	0.8150	N/A
	RF	0.8181	0.38%
	XGB	0.8166	0.20%
	LR	0.8191	0.50%
EEG Eye State	MV	0.9446	N/A
	RF	0.9617	1.81%
	XGB	0.9524	0.83%
	LR	0.9526	0.85%

Note: N/A = there was no data available

RF, LR and XGB ensembles are compared with MV ensemble in table 32. The accuracy difference between them is reported in percentage. MV ensemble was chosen as the base of comparison since it had the smallest ensemble accuracy in this experiment. The improvement of the RF, LR and XGB ensembles over MV ensemble is shown here. It shows that RF ensemble has 4.21%, 0.38%, and 1.81% better performance than MV on the Adult, Credit Card Clients, and EEG Eye State data sets. XGB ensemble has 2.78%, 0.20%, and 0.83% better performance than MV ensemble on the three data sets. LR ensemble has 5.02%, 0.50%, and 0.85% better performance than MV ensemble on the three data sets. Based on values in table 32, LR ensemble is the best and most stable method of combining the selected base models.

However, the other model selection method, Cramér's V correlation analysis, decreased the classification accuracy of RF, XGB, and MV on all three data sets. There are two reasons that might cause the poor ensemble performance. One reason is that Cramér's V correlation analysis doesn't fit those three ensemble methods. Another reason might be that the selected base models present only two thirds of all base models. Less accuracy is the expected result since a smaller number of base models were ensembled.

Experiment Four: Ensemble Selected Base Models and MCA Factor Scores

In addition to combining selected base models as in experiment three, factor scores of multiple correspondence analysis were added and the selected base models were ensembled by three different ensemble methods, XGB, RF, and LR. LR ensemble method combined MCA factor scores with base models selected by backward AIC method. XGB and RF ensemble method combined base models selected by Cramér's

V analysis with MCA factor scores. Base models selected by the Cramér's V correlation analysis are still the same twenty-one ERT and XGB base models which are selected in experiment three. The backward selection method in experiment four initially combines not only all base models but also MCA factor scores. The selection procedure is almost the same as that in experiment three. The eliminated attribute in each backward step can be a base model or a MCA factor score. However, it turns out that the final selected base models are the same as those in experiment three. As a result, the classification accuracies of LR ensemble are the same as those in experiment three. Majority voting ensemble is not applicable in experiment four since non-categorical MCA factor scores are involved in the experiment.

Ensembles Compared with Individual Base Model

As in experiment three, logistic regression is the only ensemble method that integrated with backward selection in experiment four. Since it performs the same as experiment three that ensembles only the selected base model, the comparison between the ensemble models and individual base models is the same as in experiment three. Based on these observation, we can conclude that MCA factor scores doesn't help LR ensemble at all on improving model performance when combined with the selected base models. The performance of LR ensemble and its comparison with base models can be found in table 30.

The ensemble performance of XGB method is better than all the thirty-one base models on the Adult and EEG Eye State data sets, and twenty-two base models on the Credit Card Clients data set. The data in table 31 show that the XGB ensemble method increased the classification accuracy by 0.73% to 10.49% from base models on the

EEG Eye State data set. It has the increased classification accuracies by 0.12% to 4.11% on the Adult data set. On the Credit Card Clients data set, it has better performance from 0.17% to 0.99% than those of base models except that nine XGB base models have better performance than the ensemble method. XGB ensemble with Cramér's V model selection and MCA factor scores is not an ideal ensemble method. Table 33 summarizes the comparison in detail.

Table 33

XGB Ensemble in Experiment Four Compared with Base Models

Data Set	Base Model	XGB Ensemble vs. RF Base Model		XGB Ensemble vs. ERT Base Model		XGB Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
Adult	1	0.8644	1.57%	0.8450	3.91%	0.8706	0.84%
	2	0.8649	1.51%	0.8433	4.11%	0.8728	0.60%
	3	0.8640	1.63%	0.8446	3.95%	0.8708	0.83%
	4	0.8646	1.55%	0.8452	3.89%	0.8730	0.57%
	5	0.8651	1.49%	0.8452	3.89%	0.8745	0.41%
	6	0.8642	1.60%	0.8445	3.96%	0.8762	0.21%
	7	0.8649	1.51%	0.8455	3.84%	0.8751	0.33%
	8	0.8649	1.52%	0.8458	3.81%	0.8770	0.12%
	9	0.8642	1.60%	0.8450	3.91%	0.8767	0.15%
	10	0.8651	1.50%	0.8458	3.80%	0.8755	0.29%
	11	N/A	N/A	0.8456	3.83%	N/A	N/A
	Average	0.8646	1.55%	0.8450	3.90%	0.8742	0.43%
Credit Card Clients	1	0.8143	0.71%	0.8121	0.99%	0.8169	0.39%
	2	0.8170	0.38%	0.8143	0.71%	0.8203	-0.02%
	3	0.8176	0.31%	0.8134	0.82%	0.8234	-0.40%
	4	0.8174	0.33%	0.8130	0.87%	0.8236	-0.42%
	5	0.8172	0.35%	0.8130	0.87%	0.8254	-0.64%
	6	0.8156	0.55%	0.8130	0.87%	0.8262	-0.74%
	7	0.8150	0.63%	0.8130	0.87%	0.8257	-0.68%
	8	0.8176	0.31%	0.8148	0.65%	0.8258	-0.69%
	9	0.8187	0.17%	0.8148	0.65%	0.8260	-0.71%
	10	0.8169	0.39%	0.8148	0.65%	0.8256	-0.67%

Data Set	Base Model	XGB Ensemble vs. RF Base Model		XGB Ensemble vs. ERT Base Model		XGB Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
	11	N/A	N/A	0.8148	0.65%	N/A	N/A
	Average	0.8167	0.41%	0.8137	0.78%	0.8239	-0.46%
EEG Eye State	1	0.9243	3.23%	0.9372	1.81%	0.9009	5.92%
	2	0.9268	2.96%	0.9424	1.25%	0.9059	5.33%
	3	0.9237	3.30%	0.9446	1.02%	0.9003	5.99%
	4	0.9295	2.66%	0.9455	0.92%	0.9119	4.64%
	5	0.9292	2.69%	0.9439	1.09%	0.9105	4.80%
	6	0.9308	2.51%	0.9468	0.78%	0.9089	4.98%
	7	0.9288	2.73%	0.9435	1.13%	0.9061	5.31%
	8	0.9306	2.54%	0.9453	0.94%	0.8988	6.16%
	9	0.9299	2.61%	0.9450	0.97%	0.8925	6.91%
	10	0.9288	2.73%	0.9473	0.73%	0.8636	10.49%
	11	N/A	N/A	0.9473	0.73%	N/A	N/A
	Average	0.9282	2.80%	0.9444	1.03%	0.8999	6.03%

Note: N/A = there was no data available

The ensemble performance of RF method is better than the performances of all the thirty-one base models on all three data sets. The RF ensemble increased the classification accuracy from 2.72% to 6.82% when compared with the base models on the Adult data set. It increased the classification accuracy from 1.05% to 3.31% on the Credit Card Clients data set. It also improved the classification accuracy from 2.93% to 12.91% on the EEG Eye State data set. Overall, the increased classification accuracies in experiment four reached the highest record in our research. Table 34 lists the comparisons in detail. Here, we notice that the combination of Cramér's V model selection and MCA factor scores helps to increase the classification accuracy of RF ensemble method.

Table 34

RF Ensemble in Experiment Four Compared with Base Models

Data Set	Base Model	RF Ensemble vs. RF Base Model		RF Ensemble vs. ERT Base Model		RF Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
Adult	1	0.8644	4.21%	0.8450	6.61%	0.8706	3.46%
	2	0.8649	4.15%	0.8433	6.82%	0.8728	3.21%
	3	0.8640	4.27%	0.8446	6.65%	0.8708	3.45%
	4	0.8646	4.18%	0.8452	6.58%	0.8730	3.19%
	5	0.8651	4.12%	0.8452	6.58%	0.8745	3.01%
	6	0.8642	4.24%	0.8445	6.66%	0.8762	2.81%
	7	0.8649	4.15%	0.8455	6.54%	0.8751	2.94%
	8	0.8649	4.15%	0.8458	6.51%	0.8770	2.72%
	9	0.8642	4.24%	0.8450	6.61%	0.8767	2.75%
	10	0.8651	4.13%	0.8458	6.50%	0.8755	2.89%
	11	N/A	N/A	0.8456	6.52%	N/A	N/A
	Average	0.8646	4.18%	0.8450	6.60%	0.8742	3.04%
Credit Card Clients	1	0.8143	3.03%	0.8121	3.31%	0.8169	2.71%
	2	0.8170	2.69%	0.8143	3.03%	0.8203	2.28%
	3	0.8176	2.62%	0.8134	3.15%	0.8234	1.89%
	4	0.8174	2.64%	0.8130	3.20%	0.8236	1.87%
	5	0.8172	2.67%	0.8130	3.20%	0.8254	1.65%
	6	0.8156	2.87%	0.8130	3.20%	0.8262	1.55%
	7	0.8150	2.94%	0.8130	3.20%	0.8257	1.61%
	8	0.8176	2.62%	0.8148	2.97%	0.8258	1.60%
	9	0.8187	2.48%	0.8148	2.97%	0.8260	1.57%
	10	0.8169	2.71%	0.8148	2.97%	0.8256	1.62%
	11	N/A	N/A	0.8148	2.97%	N/A	N/A
	Average	0.8167	2.73%	0.8137	3.11%	0.8239	1.83%
EEG Eye State	1	0.9243	5.50%	0.9372	4.04%	0.9009	8.24%
	2	0.9268	5.21%	0.9424	3.47%	0.9059	7.64%
	3	0.9237	5.56%	0.9446	3.23%	0.9003	8.31%
	4	0.9295	4.91%	0.9455	3.13%	0.9119	6.93%
	5	0.9292	4.94%	0.9439	3.31%	0.9105	7.10%
	6	0.9308	4.76%	0.9468	2.99%	0.9089	7.28%
	7	0.9288	4.98%	0.9435	3.35%	0.9061	7.62%

Data Set	Base Model	RF Ensemble vs. RF Base Model		RF Ensemble vs. ERT Base Model		RF Ensemble vs. XGB Base Model	
		Base Model	Accuracy Increase	Base Model	Accuracy Increase	Base Model	Accuracy Increase
	8	0.9306	4.78%	0.9453	3.15%	0.8988	8.49%
	9	0.9299	4.86%	0.9450	3.19%	0.8925	9.25%
	10	0.9288	4.98%	0.9473	2.93%	0.8636	12.91%
	11	N/A	N/A	0.9473	2.93%	N/A	N/A
	Average	0.9282	5.05%	0.9444	3.25%	0.8999	8.35%

Note: N/A = there was no data available

Comparison of Ensemble Methods in Experiment Four

Combining selected base models and MCA factor scores, RF ensemble method with Cramér's V selected base models perform extremely well on all the data sets. It provides the best accuracy on all three data sets. It outperforms LR ensemble with backward selected base models and XGB ensemble with Cramér's V model selection. XGB ensemble has worse classification performance than LR ensemble on the Adult data set, but performs better than LR ensemble on the Credit Card Clients and EEG Eye State data sets.

RF and XGB ensembles are compared with LR ensemble in table 35. The accuracy difference between them is reported in percentage. LR ensemble is chosen as the comparison base since it has the smallest ensemble accuracy on two data sets in this experiment, and its performance is stable in all four experiments. The accuracy difference in percentage between RF ensemble, XGB ensemble and LR ensemble is reported in table 35. It shows that RF ensemble has 0.63%, 2.43%, and 2.36% better performance than LR on the Adult, Credit Card Clients, and EEG Eye State data sets respectively. XGB ensemble has 0.12% and 0.17% better performance than LR

ensemble on the Credit Card Clients and EEG Eye State data sets; but -1.92% worse performance on the Adult data set. In summary, RF ensemble method performs the best and provides the best classification accuracies on all data sets. XGB ensemble performs relatively better than LR ensemble.

Table 35

Ensemble Comparison in Experiment Four

Data Set	Ensemble Method	Ensemble Accuracy	Accuracy Increased from MV Ensemble
Adult	LR	0.8952	N/A
	RF	0.9008	0.63%
	XGB	0.8780	-1.92%
Credit Card Clients	LR	0.8191	N/A
	RF	0.8390	2.43%
	XGB	0.8201	0.12%
EEG Eye State	LR	0.9526	N/A
	RF	0.9751	2.36%
	XGB	0.9542	0.17%

Note: N/A = there was no data available

Ensembles Compared with Benchmarks, Experiment One, Two, and Three

Table 36 summarizes the classification accuracy of benchmarks and ensemble models in experiment one, two, three, and four. Table 37 compares the ensemble methods in experiment four to benchmarks and the same type of ensemble methods in experiment one, two, and three. It shows that the performance of LR ensemble is very stable. It has the same predictive accuracy on Adult data set in all four experiments. On the Credit Card Clients and EEG Eye State data sets, LR has the same classification accuracies in experiment one and two, and has the same classification performance in experiment three and four. Its classification accuracies in experiment three and four are

a little bit higher than those in experiment one and two. XGB ensemble performs the best on all three data sets in experiment two that ensemble all base models and MCA factor scores. RF ensemble achieved the best performance on all three data sets in experiment four that combines only selected base models and MCA factor scores. The second best performance of RF ensemble happened in experiment two that integrates all base models and MCA factor scores. RF ensemble also achieved good performance in experiment one that combines all base models.

Table 36

Ensemble Accuracy of Experiment One, Two, Three, and Four

Data Set	BMK Accuracy	Ensemble Method	Ensemble Accuracy			
			Exp 1	Exp2	Exp3	Exp4
Adult	0.8646	RF	0.8957	0.8996	0.8883	0.9008
		XGB	0.8953	0.8953	0.8761	0.8780
		LR	0.8952	0.8952	0.8952	0.8952
Credit Card Clients	0.8167	RF	0.8360	0.8398	0.8181	0.8390
		XGB	0.8217	0.8224	0.8166	0.8201
		LR	0.8194	0.8194	0.8191	0.8191
EEG Eye State	0.9282	RF	0.9671	0.9733	0.9617	0.9751
		XGB	0.9539	0.9544	0.9524	0.9542
		LR	0.9522	0.9522	0.9526	0.9526

Note: BMK = Benchmark; Exp = Experiment Design; MV = Majority Voting; RF = Random Forest; XGB = Extreme Gradient Boosting; LR =Logistic Regression

Compared with benchmarks, all the ensemble methods in experiment four outperform the benchmarks on all three data sets. RF ensemble method increased 2.73%, 4.19%, and 5.05% accuracies from benchmarks. LR ensemble method increased the accuracy by 0.29%, 2.63% and 3.54% on all three data sets. XGB ensemble method increased the accuracy by 0.42%, 1.55% and 2.80% respectively. Table 37 shows the detail of comparison.

Table 37

Experiment Four Compared to Benchmarks, Experiment One, Two, and Three

Data Set	Ensemble Method	Accuracy Increase			
		Exp 4 vs. BMK	Exp4 vs. Exp1	Exp4 vs. Exp2	Exp4 vs. Exp3
Adult	RF	4.19%	0.57%	0.13%	1.41%
	XGB	1.55%	-1.93%	-1.93%	0.22%
	LR	3.54%	0.00%	0.00%	0.00%
Credit Card Clients	RF	2.73%	0.36%	-0.10%	2.55%
	XGB	0.42%	-0.19%	-0.28%	0.43%
	LR	0.29%	-0.04%	-0.04%	0.00%
EEG Eye State	RF	5.05%	0.83%	0.18%	1.39%
	XGB	2.80%	0.03%	-0.02%	0.19%
	LR	2.63%	0.04%	0.04%	0.00%

Note: BMK = Benchmark; Exp = Experiment Design; MV = Majority Voting; RF = Random Forest; XGB = Extreme Gradient Boosting; LR =Logistic Regression

Compared with the same ensemble methods in experiment one, LR ensemble method has the same or comparable accuracies on all three data sets. It achieved a 0.04% increased accuracy on the EEG Eye State data set, and a 0.04% decreased accuracy on the Credit Card Client data set which is almost the same accuracy as that in experiment one. XGB ensemble method has -1.93% and -0.19% decreased accuracy on the Adult, Credit Card Client, and 0.03% increased accuracy on the EEG Eye Status data set. RF ensemble method increased the accuracy by 0.57%, 0.36%, and 0.83% on the three data sets respectively. Overall, compared with the same ensemble method in experiment one which ensembles all base models, three ensembles in experiment four that had model selection and MCA factor scores are defeated in accuracy. They are two XGB ensemble methods that combines twenty-one selected base models and MCA factor scores on the Adult and Credit Card Client data sets, and one LR ensemble with AIC backward selection and MCA factor scores on the Credit Card Client data set.

Compared with the same ensemble methods in experiment two, LR ensemble method has the same or comparable accuracies on all three data sets. It achieved a 0.04% increased accuracy on EEG Eye State data set, and a -0.04% decreased accuracy on Credit Card Client data set; it has the same accuracy as that on the Adult data set in experiment two. XGB ensemble method has all decreased -1.93%, -0.28%, and -0.02% accuracy on the Adult, Credit Card Client, and EEG Eye Status data sets. RF ensemble method decreased the accuracy by 0.10% on the Credit Card Client data set, and increased the accuracy by 0.13% and 0.18% on the Adult and EEG Eye State data sets. In summary, compared with the same ensemble method in experiment two which ensembles all base models and MCA factor scores, five ensembles with model selection are defeated in accuracy. Only the RF ensembles with twenty-one selected base models and MCA factor scores on the Adult and EEG Eye State data sets achieved better accuracies than the same ensemble methods with all thirty-one base models and MCA factor scores in experiment two. It shows that the random forest ensemble works better on selected optimal subsets when MCA factor scores are used in the ensemble.

Compared with the same ensemble methods in experiment three, LR ensemble has the same accuracies on all three data sets. XGB ensemble method has 0.22%, 0.43%, and 0.19% increased accuracy on the Adult, Credit Card Client, and EEG Eye Status data sets. RF ensemble method increased the accuracy by 1.41%, 2.55%, and 1.39% on the three data sets respectively. In summary, except for the LR ensemble method which has the same performance, XGB and RF ensemble methods with only selected models and MCA factor scores outperform the same ensemble methods with

selected base models only. Hence, we conclude that involving the MCA factor scores in ensembles improves the performance of classification.

Experiment four shows that ensemble methods combining selected base model and MCA factor scores outperform the same ensemble methods in experiment three that combines only selected base models. They also outperform the same ensemble methods in experiment one that integrates all thirty-one base models except for the XGB ensembles on two data sets and one LR ensemble on one data set. However, they are defeated by the same ensemble methods in experiment two that ensemble all base model and MCA factor scores except for RF ensemble method on two data sets. Among all the ensemble methods in the four experiments, RF ensemble models almost always achieve the best classification accuracy in each experiment. They work exceptionally well when MCA factor scores are added and model selection is used. LR ensemble method has stable and acceptable classification accuracy in all the experiments. Combining base model selection procedure or adding MCA factors scores doesn't affect the performance of LR ensemble at all or improves its performance very little. XGB ensemble performs the second best when compared with the other two ensemble methods. Adding MCA factor scores in ensemble improves its performance when combining either all base models or selected base models. However, base model selection decreases its performance a lot on all three test data sets.

Chapter 5

Conclusions and Summary

As a proven effective approach, ensemble models have been applied in numerous classification tasks (Zhang et al., 2011). In this research, the ensemble model was used to combine the predictions of three types of base models to achieve higher out-of-sample classification accuracy than the base models. Specifically, the base models themselves are ensemble-based models. They are random forest, extreme gradient boosting, and extremely randomized trees model that always provide the best performance in various situations. The literature shows that ensemble classifiers are particularly effective when their constituent base models are diverse in terms of their prediction accuracy in different regions of the feature space (Dietterich, 2001). Randomization of the three ensemble-based base classifiers provided enough diversity to ensure the success of further ensemble approach in this research. The research investigated methods of integrating ensemble models by treating them as base models in four designed experiments. Strategies for combining ensemble classifiers that resulted in higher classification accuracy than its constituent ensemble modes were identified. Various strategies were evaluated using three public domain data sets which have been extensively used for benchmarking classification models.

Random forest, extremely randomized trees, and extreme gradient boosting model were generated as base models due to their high predictive accuracy and high diversity resulting from randomization (Brieman, 2001; Geurts, Ernst, & Wehenkel, 2006; Friedman, 2001). They are all tree-based and ensemble-based machine learning algorithms that utilize the subsample of training data set, subset of attributes, and/or random cut-point choice when growing tsrees to create enough diversity. Adjusting the parameter of extreme gradient boosting model, the number of trees of random forest and extremely randomized trees when using R packages to create base models also contributes diversity to the structure of base models. It was noticed that there is no linear relationship between the number of trees or parameter settings with the performance of base models. It is hard to conclude which type of base models performs the best because they perform differently on different data sets. Extreme gradient boosting model outperforms the other two types of base models in producing average classification accuracy on two data sets, but has the smallest average classification accuracy on the third data set. Extremely randomized trees is outperformed by the other two types of base models in providing average classification accuracy on two data sets. However, it provides the best classification accuracy on the third data set. The performance of random forest base models on all three data sets is between those of extreme gradient boosting and extremely randomized trees base models.

The literature shows that majority voting is mostly used as benchmarking ensemble method in many researches (Brieman, 2001). Logistic regression has also proven to be a good ensemble method in stacking (Wolpert, 1992). In addition to majority voting and logistic regression ensemble, we introduced random forest and

extreme gradient boosting model as additional ensemble methods to our research. In experiment one, all thirty-one base models are combined by these four different ensemble methods: majority voting, random forest, extreme gradient boosting, and logistic regression. Except for majority voting, all other three ensemble methods are machine learning algorithms or statistical models. The performance of ensemble methods was compared with that of base models. Majority voting ensemble achieved better performance than 72% of individual base models on the three data sets. It outperformed 83% of random forest base models, 67% extremely randomized trees base models, and 32% of extreme gradient boosting base models on the three test data sets. Extreme gradient boosting ensemble achieved better performance than 92% of base models. Only eight extreme gradient boosting base models defeated it. Logistic regression ensemble achieved better performance than 90% of base models. Only nine extreme gradient boosting base model had better performance than it. Random forest ensemble outperformed all of the base models. Compared with benchmarks, which are the average accuracy of random forest base models, all four ensembles achieved improved performance. Random forest ensemble achieved the best classification accuracy among the four ensemble methods. Logistic regression had a comparable classification accuracy as the extreme gradient boosting ensemble. Majority voting ensemble performed the worst among the four ensemble approaches. It is concluded that random forest ensemble is the best ensemble method when combining all three kinds of base models. Majority voting method is not an ideal ensemble method in this experiment.

It is very common that all base models are combined together for the final output in lots of research. However, researchers have also showed that combining base models with some desirable characteristics worked better than only combining all base models (Rodríguez, Kuncheva, & Alonso, 2006). So, factor scores of multiple correspondence analysis (MCA) were generated and combined with base models in our experiments. The MCA factor scores are designed to preserve the variability information in the data and capture the new features of base models. In experiment two, extreme gradient boosting, random forest, and logistic regression model were used to combine all base models and MCA factor scores. Compared with individual base models, logistic regression ensemble achieved better performance than 90% of base models. Only nine extreme gradient boosting base models had better performance than logistic regression ensemble. Extreme gradient boosting ensemble achieved better performance than 92% of base models. Only eight extreme gradient boosting base models outperformed it. Random forest ensemble outperformed all of the base models. Compared with benchmarks, all three ensembles achieved improved performance. Again, random forest ensemble achieved the best performance among the three ensemble methods. Logistic regression had a comparable classification accuracy as the extreme gradient boosting ensemble. Compared with the same ensemble in experiment one, random forest ensemble improved the accuracy the most; extreme gradient boosting had a minor performance improvement or similar performance; logistic regression produced the same accuracy on all data sets. It is concluded that random forest ensemble is the best ensemble method when combining all of the three kinds of

base models with MCA factor scores. MCA factor scores help the random forest ensemble method in providing more accurate predictions.

Selecting a subset of base models might improve the accuracy of the sfinal decision (Tsoumakas, Partalas, & Vlahavas, 2008). The criterion of selecting the base models can be based on accuracy and diversity (Hu, 2001). The three kinds of base models in the experiments in this research are already ensemble-based models and proven to be able to achieve high accuracy in most situations, hence the focus is on choosing base models with the most diversity in this research. Cramér's V correlation analysis and backward selection were applied in experiment three and four to choose the optimal subset of diverse base models to ensemble (Abdelazeem, 2008). Backward selection associated with logistic regression ensemble method selected less than 50% of base models in the final ensemble. The selected base models are a mix of three kinds of base models. Two data sets favored larger numbers of extreme gradient boosting base models in the final ensemble; and one data set favored a larger number of random forest models in the final ensemble. After applying another model selection method, Cramér's V correlation analysis, ten extreme gradient boosting and eleven extremely randomized trees base models were chosen by evaluating the average of correlation coefficients between them. These two types of models were selected because they had the smallest correlation on average. These twenty-one base models were combined with majority voting, random forest, and extreme gradient boosting in experiment three.

In experiment three, only selected base models were ensembled. Compared with individual base models, majority voting ensemble achieved better performance than only 49% of base models. Extreme gradient boosting ensemble achieved better

performance than only 78% of base models; thirteen extreme gradient boosting and seven random forest base models outperformed extreme gradient boosting ensemble. Random forest ensemble did not outperform all the base models as in experiment two; nine extreme gradient boosting base models had better performance than random forest ensemble. Logistic regression outperformed 90% of base models; the same nine extreme gradient boosting base models had better performance than logistic regression ensemble. Among the ensemble methods in experiment three, random forest ensemble models had the best performance. Although logistic regression ensemble only integrated around 33% to 50% base models, it achieved about the same accuracy as the ensemble methods that integrated all base models. Compared with benchmarks, majority voting ensemble outperformed the benchmark only on one data set; extreme gradient boosting ensemble outperformed the benchmark on two data sets; random forest and logistic regression outperformed the benchmarks on all three data sets. Compared with the same ensemble methods in experiment two, logistic regression ensemble had the same or very close accuracy as that in experiment two. Extreme gradient boosting and random forest ensemble both had worse performance than they had in experiment two. According to the results in experiment one, two, and three, the three ensemble methods in experiment two which combined all base models and MCA factor scores achieved better classification accuracy than the same methods in experiment one which combined all base models and in experiment three which integrated only selected base models.

Since the ensembles in experiment three had all decreased performance when compared with those ensembles in experiment one and two, MCA factor scores were

added to combine with the selected base models in experiment four. Whether the MCA factor scores could improve the ensemble performance was evaluated. Compared with base models, extreme gradient boosting ensemble achieved better performance than 93% of base models with only nine extreme gradient boosting base models outperforming it. The random forest ensemble outperformed all the base models as in experiment two. Logistic regression ensemble had the same performance as in experiment three.

Among the ensemble methods in experiment four, random forest ensemble outperformed a larger number of base models than extreme gradient boosting and logistic regression ensemble. Logistic regression ensemble achieved the same performance as it did in experiment three, which supports the conclusion that adding MCA factor scores doesn't help to increase logistic regression ensemble at all. Logistic regression, extreme gradient boosting and random forest ensemble outperformed the benchmarks on three data sets. Compared with the same ensemble methods in experiment one, random forest ensemble had better performance in experiment four; extreme gradient boosting ensemble had worse performance on two data sets and a comparable performance on the third data set. When compared with the same ensemble methods in experiment two, five ensembles were defeated in experiment four. It seemed that ensembles in experiment two had better performance in overall. However, it was noticed that random forest in experiment four outperformed themselves in experiment two on two data sets. Compared with experiment three, it is not surprising that except for logistic regression that had stable performance in all the four experiments, ensembles in experiment four performed better than in experiment three.

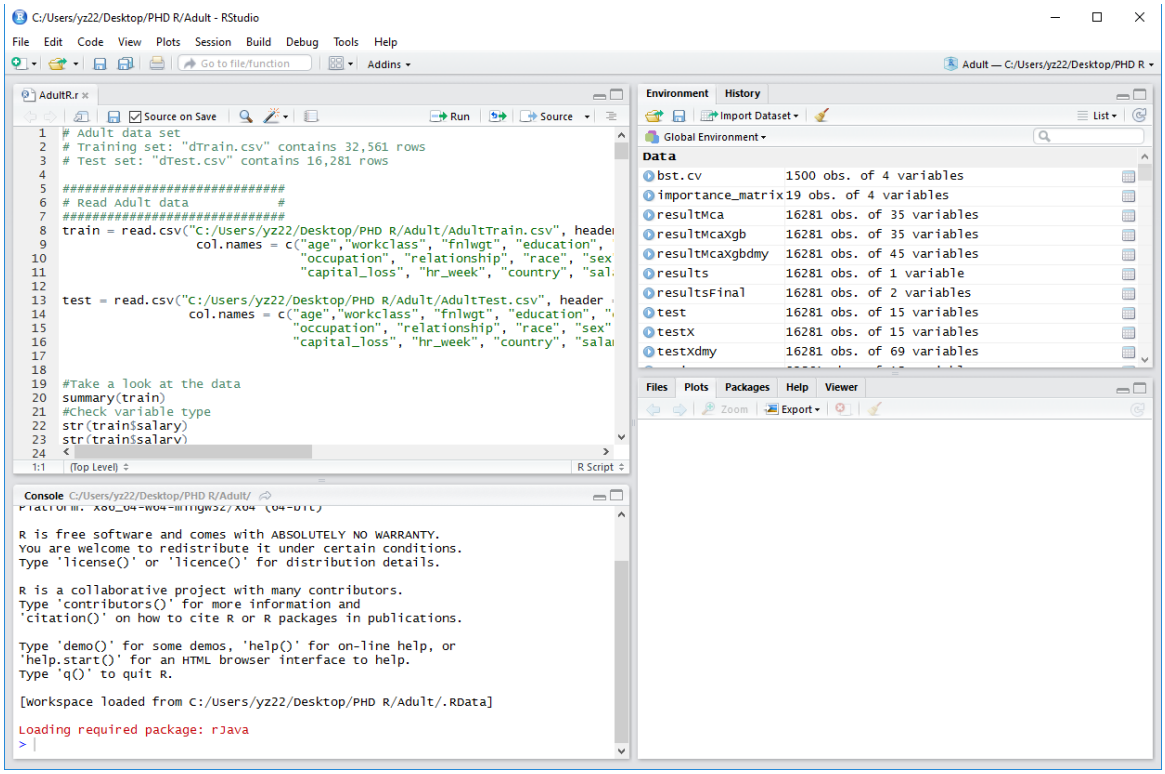
MCA factor scores helped a little bit in the extreme gradient boosting ensemble but a lot in the random forest ensemble.

As an ensemble method of three kinds of ensemble-based high performance base models, majority voting performed the worst in combining either all base models or selected base models. It is not an ideal method to do the final combination. Another ensemble method, logistic regression, achieved competitive but not the best performance when compared with other ensemble approaches. It is not sensitive to model selection or MCA factor scores since it produced very stable classification accuracies regardless of the model selection applied or MCA factor scores presented. The third ensemble method, extreme gradient boosting, did a better job when more variables, here more base models, were combined. In our experiment, since MCA factor scores were treated the same as base models in ensemble, extreme gradient boosting ensemble performed better when MCA factor scores were involved; however, it performed poorly when model selection was involved. Extreme gradient boosting ensemble is sensitive to the number of inputs, which include both base models and MCA factor scores, to the final ensemble approach. The more base models included as inputs, the better performance extreme gradient boosting ensemble can achieve. The fourth ensemble, random forest, is the most successful method of combining the three kinds of high performance tree-based ensemble models. It performed the best when combining all base models, all base models with MCA factor scores, or selected base models with MCA factor scores when compared with other types of ensemble methods. Especially, it achieved the highest classification accuracy on two data sets when combining only twenty-one selected base models and MCA factor scores. Adding

MCA factor scores definitely helped to improve random forest ensemble method. On the contrary, applying only model selection decreased its performance. However, applying both model selection and MCA factor scores worked extremely successfully in improving its ensemble performance. Overall, ensemble methods among random forest, extreme gradient boosting, and extremely randomized trees, the best approach to ensemble tree-based ensemble models is random forest ensemble method with or without model selection, and then combining MCA factor scores with those selected or all base models.

The research is designed to investigate if there is an approach which can integrate ensemble-based models to achieve even better classification accuracy. The experiments are limited to binary classification problems. Future research can be extended to multiple classification problems, or even further to numerical predictions. It is expected that the findings of this research can be applied in the real world since all the testing data sets are real but not simulated data sets. The three data sets are collected from the field of finance, national survey, and physics respectively. Future research can be applied to other types of real world data.

Appendix A: RStudio Interface



Appendix B: R Code of Experiments on EEG Eye State Data Set

```
# EGG data set
# data set: "EEG-DATA.cvs" contains 14,980 rows

################################
# Read EGG data                #
################################
whole = read.csv("C:/Users/yz22/Desktop/PHD R/EEG/EEG-DATA.csv", header = FALSE,
            col.names = c("AF3","F7", "F3", "FC5", "T7", "P7",
                          "O1", "O2", "P8", "T8", "FC6",
                          "F4", "F8", "AF4", "eyeDetection"))

###Data partition into 70% and 30%###
data(whole)
n = nrow(whole)
trainIndex = sample(1:n, size = round(0.7*n), replace=FALSE)
train = whole[trainIndex ,]
test = whole[-trainIndex ,]

# data frame for predicted results on test data

results = data.frame(y = test$eyeDetection)

################################
# End Read EGG data  #
################################

#################################################################
# eXtreme Gradient Boosting: install.packages("xgboost") #
#################################################################
install.packages("drat", repos="https://cran.rstudio.com")
drat:::addRepo("dmlc")
install.packages("xgboost", repos="http://dmlc.ml/drat/", type = "source")

require(xgboost)
set.seed(12345)

trainX = train
testX = test

##rename the target variable into y##
names(trainX)[15]<-"y"
names(testX)[15]<-"y"

#binarize all factors in train data set
library(caret)
dmy <- dummyVars(" ~ .", data=trainX)
trainXdmy <- data.frame(predict(dmy, newdata=trainX))
dim(trainXdmy)
names(trainXdmy)
```

```
#binarize all factors in test data set
dmyTest <- dummyVars(" ~ .", data=testX)
testXdmy <- data.frame(predict(dmyTest, newdata=testX))
dim(testXdmy)
names(testXdmy)

##############################################
#prepared a varialbe of target, and a matrix for predictor#
outcomeName <-c('y')

predictors <- names(trainXdmy)[!names(trainXdmy)%in% outcomeName]
predictorsTest <- names(trainXdmy)[!names(trainXdmy)%in% outcomeName]

#train
#nrounds parameter is adjustable

library(xgboost)
#For variable importance
library(DiagrammeR)
library(Ckmeans.1d.dp)
set.seed(12345)

####################
#xgboost 1st model##
####################

#run 10 fold cross validation and choose the best round;round 80 to 120 all have the relative
similar accuracy
#set up parameters for Xgboost#
param <- list("objective" = "binary:logistic",
              "eval_metric" = "logloss",
              "eta" = 0.1, "max.depth" = 2)

bst.cv <- xgb.cv(param=param, data = as.matrix(trainXdmy[,predictors]), label =
trainXdmy[,outcomeName], nfold=10, nround = 1500)
plot(log(bst.cv$test.logloss.mean),type = "l")

# round 45 have the highest accuracy#
bst <- xgboost(data = as.matrix(trainXdmy[,predictors]), label = trainXdmy[,outcomeName],
max.depth = 2, eta = 1, nround = 1068,
               nthread = 2, objective ="binary:logistic")
gc()
#make prediction#
predictions <- predict(bst, as.matrix(testXdmy[,predictorsTest]), outputmargin= FALSE)
#outputmargin has to be FALSE to produce probability
#predictions[1:10]
print(-mean(log(predictions)*testXdmy$y+log(1-predictions)*(1-testXdmy$y)))

# Get the variable importance
importance_matrix <- xgb.importance(predictors, model = bst)
```

```
xgb.plot.importance(importance_matrix[1:10])
xgb.plot.tree(feature_names =predictors, model = bst, n_first_tree = 2)

#convert to categorical target and caculate the accuracy
yPredCat1 <- ifelse(predictions <= 0.5,0,1)
yPredCat1[1:10]
results$xgb1 = yPredCat1
confusionXgb1=table(yPredCat1,testXdmy$y)
accuracyXgb1 = sum(diag(confusionXgb1))/sum(confusionXgb1)
cat("xgboost Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyXgb1))
confusionXgb1

####################
#xgboost 2nd model##
####################

param <- list("objective" = "binary:logistic",
              "eval_metric" = "logloss",
              "eta" = 0.9, "max.depth" = 2)
#run 10 fold cross validation and choose the best round;round 80 to 120 all have the relative
similar accuracy
bst.cv <- xgb.cv(param=param, data = as.matrix(trainXdmy[,predictors]), label =
trainXdmy[,outcomeName], nfold=10, nround = 1500)
plot(log(bst.cv$test.logloss.mean),type = "l")

bst <- xgboost(data = as.matrix(trainXdmy[,predictors]), label = trainXdmy[,outcomeName],
max.depth = 2, eta = 0.9, nround = 850,
               nthread = 2, objective ="binary:logistic")
gc()
#make prediction#
predictions <- predict(bst, as.matrix(testXdmy[,predictorsTest]), outputmargin= FALSE)
#outputmargin has to be FALSE to produce probability
#predictions[1:10]
print(-mean(log(predictions)*testXdmy$y+log(1-predictions)*(1-testXdmy$y)))

# Get the variable importance
importance_matrix <- xgb.importance(predictors, model = bst)
xgb.plot.importance(importance_matrix[1:10])
xgb.plot.tree(feature_names =predictors, model = bst, n_first_tree = 2)

#convert to categorical target and caculate the accuracy
yPredCat2 <- ifelse(predictions <= 0.5,0,1)
yPredCat2[1:10]
results$xgb2 = yPredCat2
confusionXgb2=table(yPredCat2,testXdmy$y)
accuracyXgb2 = sum(diag(confusionXgb2))/sum(confusionXgb2)
cat("xgboost Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyXgb2))
confusionXgb2
```

```
####################
#xgboost 3rd model##
####################
set.seed(63521)
param <- list("objective" = "binary:logistic",
              "eval_metric" = "logloss",
              "eta" = 0.8, "max.depth" = 2)
#run 10 fold cross validation and choose the best round;round 80 to 120 all have the relative
similar accuracy
bst.cv <- xgb.cv(param=param, data = as.matrix(trainXdmy[,predictors]), label =
trainXdmy[,outcomeName], nfold=10, nround = 1000)
plot(log(bst.cv$test.logloss.mean),type = "l")

bst <- xgboost(data = as.matrix(trainXdmy[,predictors]), label = trainXdmy[,outcomeName],
max.depth = 2, eta = 0.8, nround = 920,
               nthread = 2, objective ="binary:logistic")
gc()
#make prediction#
predictions <- predict(bst, as.matrix(testXdmy[,predictorsTest]), outputmargin= FALSE)
#outputmargin has to be FALSE to produce probability
#predictions[1:10]
print(-mean(log(predictions)*testXdmy$y+log(1-predictions)*(1-testXdmy$y)))

# Get the variable importance
importance_matrix <- xgb.importance(predictors, model = bst)
xgb.plot.importance(importance_matrix[1:10])
xgb.plot.tree(feature_names =predictors, model = bst, n_first_tree = 2)

#convert to categorical target and caculate the accuracy
yPredCat3 <- ifelse(predictions <= 0.5,0,1)
yPredCat3[1:10]
results$xgb3 = yPredCat3
confusionXgb3=table(yPredCat3,testXdmy$y)
accuracyXgb3 = sum(diag(confusionXgb3))/sum(confusionXgb3)
cat("xgboost Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyXgb3))
confusionXgb3

####################
#xgboost 4th model##
####################
param <- list("objective" = "binary:logistic",
              "eval_metric" = "logloss",
              "eta" = 0.7, "max.depth" = 2)
#run 10 fold cross validation and choose the best round;round 80 to 120 all have the relative
similar accuracy
bst.cv <- xgb.cv(param=param, data = as.matrix(trainXdmy[,predictors]), label =
trainXdmy[,outcomeName], nfold=10, nround = 1500)
plot(log(bst.cv$test.logloss.mean),type = "l")
```

```
bst <- xgboost(data = as.matrix(trainXdmy[,predictors]), label = trainXdmy[,outcomeName],
max.depth = 2, eta = 0.7, nround = 1427,
            nthread = 2, objective ="binary:logistic")
gc()
#make prediction#
predictions <- predict(bst, as.matrix(testXdmy[,predictorsTest]), outputmargin= FALSE)
#outputmargin has to be FALSE to produce probability
#predictions[1:10]
print(-mean(log(predictions)*testXdmy$y+log(1-predictions)*(1-testXdmy$y)))

# Get the variable importance
importance_matrix <- xgb.importance(predictors, model = bst)
xgb.plot.importance(importance_matrix[1:10])
xgb.plot.tree(feature_names =predictors, model = bst, n_first_tree = 2)

#convert to categorical target and caculate the accuracy
yPredCat4 <- ifelse(predictions <= 0.5,0,1)
yPredCat4[1:10]
results$xgb4 = yPredCat4
confusionXgb4=table(yPredCat4,testXdmy$y)
accuracyXgb4 = sum(diag(confusionXgb4))/sum(confusionXgb4)
cat("xgboost Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyXgb4))
confusionXgb4

####################
#xgboost 5th model##
####################
param <- list("objective" = "binary:logistic",
            "eval_metric" = "logloss",
            "eta" = 0.6, "max.depth" = 2)
#run 10 fold cross validation and choose the best round;
bst.cv <- xgb.cv(param=param, data = as.matrix(trainXdmy[,predictors]), label =
trainXdmy[,outcomeName], nfold=10, nround = 1500)
plot(log(bst.cv$test.logloss.mean),type = "l")

bst <- xgboost(data = as.matrix(trainXdmy[,predictors]), label = trainXdmy[,outcomeName],
max.depth = 2, eta = 0.6, nround = 1440,
            nthread = 2, objective ="binary:logistic")
gc()
#make prediction#
predictions <- predict(bst, as.matrix(testXdmy[,predictorsTest]), outputmargin= FALSE)
#outputmargin has to be FALSE to produce probability
#predictions[1:10]
print(-mean(log(predictions)*testXdmy$y+log(1-predictions)*(1-testXdmy$y)))

# Get the variable importance
importance_matrix <- xgb.importance(predictors, model = bst)
xgb.plot.importance(importance_matrix[1:10])
xgb.plot.tree(feature_names =predictors, model = bst, n_first_tree = 2)
```

```
#convert to categorical target and caculate the accuracy
yPredCat5 <- ifelse(predictions <= 0.5,0,1)
yPredCat5[1:10]
results$xgb5 = yPredCat5
confusionXgb5=table(yPredCat5,testXdmy$y)
accuracyXgb5 = sum(diag(confusionXgb5))/sum(confusionXgb5)
cat("xgboost Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyXgb5))
confusionXgb5

####################
#xgboost 6th model##
####################
param <- list("objective" = "binary:logistic",
              "eval_metric" = "logloss",
              "eta" = 0.5, "max.depth" = 2)
#run 10 fold cross validation and choose the best round;
bst.cv <- xgb.cv(param=param, data = as.matrix(trainXdmy[,predictors]), label =
trainXdmy[,outcomeName], nfold=10, nround = 1500)
plot(log(bst.cv$test.logloss.mean),type = "l")

bst <- xgboost(data = as.matrix(trainXdmy[,predictors]), label = trainXdmy[,outcomeName],
max.depth = 2, eta = 0.5, nround = 1480,
               nthread = 2, objective ="binary:logistic")
gc()
#make prediction#
predictions <- predict(bst, as.matrix(testXdmy[,predictorsTest]), outputmargin= FALSE)
#outputmargin has to be FALSE to produce probability
#predictions[1:10]
print(-mean(log(predictions)*testXdmy$y+log(1-predictions)*(1-testXdmy$y)))

# Get the variable importance
importance_matrix <- xgb.importance(predictors, model = bst)
xgb.plot.importance(importance_matrix[1:10])
xgb.plot.tree(feature_names =predictors, model = bst, n_first_tree = 2)

#convert to categorical target and caculate the accuracy
yPredCat6 <- ifelse(predictions <= 0.5,0,1)
yPredCat6[1:10]
results$xgb6 = yPredCat6
confusionXgb6=table(yPredCat6,testXdmy$y)
accuracyXgb6 = sum(diag(confusionXgb6))/sum(confusionXgb6)
cat("xgboost Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyXgb6))
confusionXgb6

####################
#xgboost 7th model##
####################
param <- list("objective" = "binary:logistic",
```

```
            "eval_metric" = "logloss",
            "eta" = 0.4, "max.depth" = 2)
#run 10 fold cross validation and choose the best round;1
bst.cv <- xgb.cv(param=param, data = as.matrix(trainXdmy[,predictors]), label =
trainXdmy[,outcomeName], nfold=10, nround = 1500)
plot(log(bst.cv$test.logloss.mean),type = "l")

bst <- xgboost(data = as.matrix(trainXdmy[,predictors]), label = trainXdmy[,outcomeName],
max.depth = 2, eta = 0.4, nround = 1500,
             nthread = 2, objective ="binary:logistic")
gc()
#make prediction#
predictions <- predict(bst, as.matrix(testXdmy[,predictorsTest]), outputmargin= FALSE)
#outputmargin has to be FALSE to produce probability
#predictions[1:10]
print(-mean(log(predictions)*testXdmy$y+log(1-predictions)*(1-testXdmy$y)))

# Get the variable importance
importance_matrix <- xgb.importance(predictors, model = bst)
xgb.plot.importance(importance_matrix[1:10])
xgb.plot.tree(feature_names =predictors, model = bst, n_first_tree = 2)

#convert to categorical target and caculate the accuracy
yPredCat7 <- ifelse(predictions <= 0.5,0,1)
yPredCat7[1:10]
results$xgb7 = yPredCat7
confusionXgb7=table(yPredCat7,testXdmy$y)
accuracyXgb7 = sum(diag(confusionXgb7))/sum(confusionXgb7)
cat("xgboost Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyXgb7))
confusionXgb7

####################
#xgboost 8th model##
####################
param <- list("objective" = "binary:logistic",
            "eval_metric" = "logloss",
            "eta" = 0.3, "max.depth" = 2)
#run 10 fold cross validation and choose the best round;
bst.cv <- xgb.cv(param=param, data = as.matrix(trainXdmy[,predictors]), label =
trainXdmy[,outcomeName], nfold=10, nround = 1500)
plot(log(bst.cv$test.logloss.mean),type = "l")

bst <- xgboost(data = as.matrix(trainXdmy[,predictors]), label = trainXdmy[,outcomeName],
max.depth = 2, eta = 0.3, nround = 1500,
             nthread = 2, objective ="binary:logistic")
gc()
#make prediction#
predictions <- predict(bst, as.matrix(testXdmy[,predictorsTest]), outputmargin= FALSE)
#outputmargin has to be FALSE to produce probability
#predictions[1:10]
```

```
print(-mean(log(predictions)*testXdmy$y+log(1-predictions)*(1-testXdmy$y)))

# Get the variable importance
importance_matrix <- xgb.importance(predictors, model = bst)
xgb.plot.importance(importance_matrix[1:10])
xgb.plot.tree(feature_names =predictors, model = bst, n_first_tree = 2)

#convert to categorical target and caculate the accuracy
yPredCat8 <- ifelse(predictions <= 0.5,0,1)
yPredCat8[1:10]
results$xgb8 = yPredCat8
confusionXgb8=table(yPredCat8,testXdmy$y)
accuracyXgb8 = sum(diag(confusionXgb8))/sum(confusionXgb8)
cat("xgboost Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyXgb8))
confusionXgb8

####################
#xgboost 9th model##
####################
param <- list("objective" = "binary:logistic",
              "eval_metric" = "logloss",
              "eta" = 0.2, "max.depth" = 2)
#run 10 fold cross validation and choose the best round;
bst.cv <- xgb.cv(param=param, data = as.matrix(trainXdmy[,predictors]), label =
trainXdmy[,outcomeName], nfold=10, nround = 1500)
plot(log(bst.cv$test.logloss.mean),type = "l")


bst <- xgboost(data = as.matrix(trainXdmy[,predictors]), label = trainXdmy[,outcomeName],
max.depth = 2, eta = 0.2, nround = 1500,
               nthread = 2, objective ="binary:logistic")
gc()
#make prediction#
predictions <- predict(bst, as.matrix(testXdmy[,predictorsTest]), outputmargin= FALSE)
#outputmargin has to be FALSE to produce probability
#predictions[1:10]
print(-mean(log(predictions)*testXdmy$salary+log(1-predictions)*(1-testXdmy$salary)))

# Get the variable importance
importance_matrix <- xgb.importance(predictors, model = bst)
xgb.plot.importance(importance_matrix[1:10])
xgb.plot.tree(feature_names =predictors, model = bst, n_first_tree = 2)

#convert to categorical target and caculate the accuracy
yPredCat9 <- ifelse(predictions <= 0.5,0,1)
yPredCat9[1:10]
results$xgb9 = yPredCat9
confusionXgb9=table(yPredCat9,testXdmy$y)
accuracyXgb9 = sum(diag(confusionXgb9))/sum(confusionXgb9)
cat("xgboost Results:\n")
```

```
cat(sprintf("Accuracy %3.4f\n", accuracyXgb9))
confusionXgb9

####################
#xgboost 10th model##
####################
param <- list("objective" = "binary:logistic",
              "eval_metric" = "logloss",
              "eta" = 0.1, "max.depth" = 2)
#run 10 fold cross validation and choose the best round;
bst.cv <- xgb.cv(param=param, data = as.matrix(trainXdmy[,predictors]), label =
trainXdmy[,outcomeName], nfold=10, nround = 1500)
plot(log(bst.cv$test.logloss.mean),type = "l")

bst <- xgboost(data = as.matrix(trainXdmy[,predictors]), label = trainXdmy[,outcomeName],
max.depth = 2, eta = 0.1, nround = 1500,
               nthread = 2, objective ="binary:logistic")
gc()
#make prediction#
predictions <- predict(bst, as.matrix(testXdmy[,predictorsTest]), outputmargin= FALSE)
#outputmargin has to be FALSE to produce probability
#predictions[1:10]
print(-mean(log(predictions)*testXdmy$y+log(1-predictions)*(1-testXdmy$y)))

# Get the variable importance
importance_matrix <- xgb.importance(predictors, model = bst)
xgb.plot.importance(importance_matrix[1:10])
xgb.plot.tree(feature_names =predictors, model = bst, n_first_tree = 2)

#convert to categorical target and caculate the accuracy
yPredCat10 <- ifelse(predictions <= 0.5,0,1)
yPredCat10[1:10]
results$xgb10 = yPredCat10
confusionXgb10=table(yPredCat10,testXdmy$y)
accuracyXgb10 = sum(diag(confusionXgb10))/sum(confusionXgb10)
cat("xgboost Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyXgb10))
confusionXgb10
################################
# End extreme random boosting #
################################

#########################
#######Extra Trees#######
#########################
#put 2g space for extra tree
options( java.parameters = "-Xmx2g" )

#package "rJava" needed to be installed before using extra trees
install.packages("rJava")
Sys.setenv(JAVA_HOME='C:\\Program Files\\Java\\jre1.8.0_101') # for 64-bit version
```

```
library(rJava)

install.packages("extraTrees")
library(extraTrees)

##############################
trainET = trainX
testET = testX

trainET=list(x=trainXdmy[,predictors],y=trainXdmy[,15])
testET=list(x=testXdmy[,predictors],y=testXdmy[,15])

##################
##1st Extra Tree     ##
##################
set.seed(13524)
et <- extraTrees(trainET$x, trainET$y, ntree=50)
yhat  <- predict(et, testET$x)
## accuracy
mean(testET$y == yhat)
## class probabilities
#convert to categorical target and caculate the accuracy
yPredET1 <- ifelse(yhat <= 0.5,0,1)
yPredET1[1:10]
results$ET1 = yPredET1
confusionET1=table(yPredET1,testXdmy$y)
accuracyET1 = sum(diag(confusionET1))/sum(confusionET1)
cat("Extra Tree Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyET1))
confusionET1

##################
##2nd Extra Tree##
##################
set.seed(54321)
et <- extraTrees(trainET$x, trainET$y, ntree=100)
yhat  <- predict(et, testET$x)
## accuracy
mean(testET$y == yhat)
## class probabilities
#convert to categorical target and caculate the accuracy
yPredET2 <- ifelse(yhat <= 0.5,0,1)
yPredET2[1:10]
results$ET2 = yPredET2
confusionET2=table(yPredET2,testXdmy$y)
accuracyET2 = sum(diag(confusionET2))/sum(confusionET2)
cat("Extra Tree Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyET2))
confusionET2

##################
```

```
##3rd Extra Tree##
##################
set.seed(12345)
et <- extraTrees(trainET$x, trainET$y, ntree=150)
yhat  <- predict(et, testET$x)
## accuracy
mean(testET$y == yhat)
## class probabilities
#convert to categorical target and caculate the accuracy
yPredET3 <- ifelse(yhat <= 0.5,0,1)
yPredET3[1:10]
results$ET3 = yPredET3
confusionET3=table(yPredET3,testXdmy$y)
accuracyET3 = sum(diag(confusionET3))/sum(confusionET3)
cat("Extra Tree Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyET3))
confusionET3

##################
##4th Extra Tree##
##################
set.seed(24635)
et <- extraTrees(trainET$x, trainET$y, ntree=200)
yhat  <- predict(et, testET$x)
## accuracy
mean(testET$y == yhat)
## class probabilities
#convert to categorical target and caculate the accuracy
yPredET4 <- ifelse(yhat <= 0.5,0,1)
yPredET4[1:10]
results$ET4 = yPredET4
confusionET4=table(yPredET4,testXdmy$y)
accuracyET4 = sum(diag(confusionET4))/sum(confusionET4)
cat("Extra Tree Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyET4))
confusionET4

##################
##5th Extra Tree##
##################
set.seed(98765)
et <- extraTrees(trainET$x, trainET$y, ntree=250)
yhat  <- predict(et, testET$x)
## accuracy
mean(testET$y == yhat)
## class probabilities
#convert to categorical target and caculate the accuracy
yPredET5 <- ifelse(yhat <= 0.5,0,1)
yPredET5[1:10]
results$ET5 = yPredET5
confusionET5=table(yPredET5,testXdmy$y)
```

```
accuracyET5 = sum(diag(confusionET5))/sum(confusionET5)
cat("Extra Tree Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyET5))
confusionET5

##################
##6th Extra Tree##
##################
set.seed(40628)
et <- extraTrees(trainET$x, trainET$y, ntree=300)
yhat  <- predict(et, testET$x)
## accuracy
mean(testET$y == yhat)
## class probabilities
#convert to categorical target and caculate the accuracy
yPredET6 <- ifelse(yhat <= 0.5,0,1)
yPredET6[1:10]
results$ET6 = yPredET6
confusionET6=table(yPredET6,testXdmy$y)
accuracyET6 = sum(diag(confusionET6))/sum(confusionET6)
cat("Extra Tree Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyET6))
confusionET6

##################
##7th Extra Tree##
##################
set.seed(59764)
et <- extraTrees(trainET$x, trainET$y, ntree=350)
yhat  <- predict(et, testET$x)
## accuracy
mean(testET$y == yhat)
## class probabilities
#convert to categorical target and caculate the accuracy
yPredET7 <- ifelse(yhat <= 0.5,0,1)
yPredET7[1:10]
results$ET7 = yPredET7
confusionET7=table(yPredET7,testXdmy$y)
accuracyET7 = sum(diag(confusionET7))/sum(confusionET7)
cat("Extra Tree Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyET7))
confusionET7

##################
##8th Extra Tree##
##################
set.seed(82604)
et <- extraTrees(trainET$x, trainET$y, ntree=400)
yhat  <- predict(et, testET$x)
## accuracy
mean(testET$y == yhat)
```

```
## class probabilities
#convert to categorical target and caculate the accuracy
yPredET8 <- ifelse(yhat <= 0.5,0,1)
yPredET8[1:10]
results$ET8 = yPredET8
confusionET8=table(yPredET8,testXdmy$y)
accuracyET8 = sum(diag(confusionET8))/sum(confusionET8)
cat("Extra Tree Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyET8))
confusionET8

##################
##9th Extra Tree##
##################
set.seed(37596)
et <- extraTrees(trainET$x, trainET$y, ntree=450)
yhat  <- predict(et, testET$x)
## accuracy
mean(testET$y == yhat)
## class probabilities
#convert to categorical target and caculate the accuracy
yPredET9 <- ifelse(yhat <= 0.5,0,1)
yPredET9[1:10]
results$ET9 = yPredET9
confusionET9=table(yPredET9,testXdmy$y)
accuracyET9 = sum(diag(confusionET9))/sum(confusionET9)
cat("Extra Tree Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyET9))
confusionET9

##################
##10th Extra Tree##
##################
set.seed(49562)
et <- extraTrees(trainET$x, trainET$y, ntree=500)
yhat  <- predict(et, testET$x)
## accuracy
mean(testET$y == yhat)
## class probabilities
#convert to categorical target and caculate the accuracy
yPredET10 <- ifelse(yhat <= 0.5,0,1)
yPredET10[1:10]
results$ET10 = yPredET10
confusionET10=table(yPredET10,testXdmy$y)
accuracyET10 = sum(diag(confusionET10))/sum(confusionET10)
cat("Extra Tree Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyET10))
confusionET10

##################
##11th Extra Tree##
```

```
##################
set.seed(84638)
et <- extraTrees(trainET$x, trainET$y, ntree=550)
yhat  <- predict(et, testET$x)
## accuracy
mean(testET$y == yhat)
## class probabilities
#convert to categorical target and caculate the accuracy
yPredET11 <- ifelse(yhat <= 0.5,0,1)
yPredET11[1:10]
results$ET11 = yPredET11
confusionET11=table(yPredET11,testXdmy$y)
accuracyET11 = sum(diag(confusionET11))/sum(confusionET11)
cat("Extra Tree Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyET11))
confusionET11

#####################################################
# randomForest: install.packages("randomForest") #
#####################################################

library(randomForest)

#create fisrt random forest#
model1 = randomForest(trainXdmy[,-15], as.factor(trainXdmy[,15]),
                xtest=testXdmy[,-15], ntree=50, set.seed(12345, kind=NULL,
normal.kind=NULL))
predicted1 = model1$test$predicted
results$FOREST1 = predicted1
confusion1 = table(predicted1, testXdmy$y)
accuracy1 = sum(diag(confusion1))/sum(confusion1)
cat("FOREST Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracy1))
confusion1

#create second random forest#
model2 = randomForest(trainXdmy[,-15], as.factor(trainXdmy[,15]),
                xtest=testXdmy[,-15], ntree=100, set.seed(54321, kind=NULL,
normal.kind=NULL))
predicted2 = model2$test$predicted
results$FOREST2 = predicted2
confusion2 = table(predicted2, testXdmy$y)
accuracy2 = sum(diag(confusion2))/sum(confusion2)
cat("FOREST Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracy2))
confusion2

#create 3rd random forest#
model3 = randomForest(trainXdmy[,-15], as.factor(trainXdmy[,15]),
                xtest=testXdmy[,-15], ntree=150, set.seed(13524, kind=NULL,
normal.kind=NULL))
```

```
predicted3 = model3$test$predicted
results$FOREST3 = predicted3
confusion3 = table(predicted3, testXdmy$y)
accuracy3 = sum(diag(confusion3))/sum(confusion3)
cat("FOREST Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracy3))
confusion3

#create 4th random forest#
model4 = randomForest(trainXdmy[,-15], as.factor(trainXdmy[,15]),
                xtest=testXdmy[,-15], ntree=200, set.seed(24531, kind=NULL,
normal.kind=NULL))
predicted4 = model4$test$predicted
results$FOREST4 = predicted4
confusion4 = table(predicted4, testXdmy$y)
accuracy4 = sum(diag(confusion4))/sum(confusion4)
cat("FOREST Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracy4))
confusion4

#create 5th random forest#
model5 = randomForest(trainXdmy[,-15], as.factor(trainXdmy[,15]),
                xtest=testXdmy[,-15], ntree=250, set.seed(31452, kind=NULL,
normal.kind=NULL))
predicted5 = model5$test$predicted
results$FOREST5 = predicted5
confusion5 = table(predicted5, testXdmy$y)
accuracy5 = sum(diag(confusion5))/sum(confusion5)
cat("FOREST Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracy5))
confusion5

#create 6th random forest#
model6 = randomForest(trainXdmy[,-15], as.factor(trainXdmy[,15]),
                xtest=testXdmy[,-15], ntree=300, set.seed(43521, kind=NULL,
normal.kind=NULL))
predicted6 = model6$test$predicted
results$FOREST6 = predicted6
confusion6 = table(predicted6, testXdmy$y)
accuracy6 = sum(diag(confusion6))/sum(confusion6)
cat("FOREST Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracy6))
confusion6

#create 7th random forest#
model7 = randomForest(trainXdmy[,-15], as.factor(trainXdmy[,15]),
                xtest=testXdmy[,-15], ntree=350, set.seed(56789, kind=NULL,
normal.kind=NULL))
predicted7 = model7$test$predicted
results$FOREST7 = predicted7
confusion7 = table(predicted7, testXdmy$y)
```

```
accuracy7 = sum(diag(confusion7))/sum(confusion7)
cat("FOREST Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracy7))
confusion7

#create 8th random forest#
model8 = randomForest(trainXdmy[,-15], as.factor(trainXdmy[,15]),
                  xtest=testXdmy[,-15], ntree=400, set.seed(98765, kind=NULL,
normal.kind=NULL))
predicted8 = model8$test$predicted
results$FOREST8 = predicted8
confusion8 = table(predicted8, testXdmy$y)
accuracy8 = sum(diag(confusion8))/sum(confusion8)
cat("FOREST Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracy8))
confusion8

#create 9th random forest#
model9 = randomForest(trainXdmy[,-15], as.factor(trainXdmy[,15]),
                  xtest=testXdmy[,-15], ntree=450, set.seed(52947, kind=NULL,
normal.kind=NULL))
predicted9 = model9$test$predicted
results$FOREST9 = predicted9
confusion9 = table(predicted9, testXdmy$y)
accuracy9 = sum(diag(confusion9))/sum(confusion9)
cat("FOREST Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracy9))
confusion9

#create 10th random forest#
model10 = randomForest(trainXdmy[,-15], as.factor(trainXdmy[,15]),
                   xtest=testXdmy[,-15], ntree=500, set.seed(69875, kind=NULL,
normal.kind=NULL))
predicted10 = model10$test$predicted
results$FOREST10 = predicted10
confusion10 = table(predicted10, testXdmy$y)
accuracy10 = sum(diag(confusion10))/sum(confusion10)
cat("FOREST Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracy10))
confusion10

################################
# End randomForest       ##############
################################

#Write predicted result in a file
results$xgb1 = yPredCat1
results$xgb2 = yPredCat2
results$xgb3 = yPredCat3
results$xgb4 = yPredCat4
results$xgb5 = yPredCat5
```

```
results$xgb6 = yPredCat6
results$xgb7 = yPredCat7
results$xgb8 = yPredCat8
results$xgb9 = yPredCat9
results$xgb10 =yPredCat10
results$ET1 = yPredET1
results$ET2 = yPredET2
results$ET3 = yPredET3
results$ET4 = yPredET4
results$ET5 = yPredET5
results$ET6 = yPredET6
results$ET7 = yPredET7
results$ET8 = yPredET8
results$ET9 = yPredET9
results$ET10 = yPredET10
results$ET11 = yPredET11
results$FOREST1 = predicted1
results$FOREST2 = predicted2
results$FOREST3 = predicted3
results$FOREST4 = predicted4
results$FOREST5 = predicted5
results$FOREST6 = predicted6
results$FOREST7 = predicted7
results$FOREST8 = predicted8
results$FOREST9 = predicted9
results$FOREST10 = predicted10

write.csv(results, "C:/Users/yz22/Desktop/PHD R/EEG/modelresults.csv")

write.csv(confusion1, "C:/Users/yz22/Desktop/PHD R/EEG/confusion1.csv")
write.csv(confusion2, "C:/Users/yz22/Desktop/PHD R/EEG/confusion2.csv")
write.csv(confusion3, "C:/Users/yz22/Desktop/PHD R/EEG/confusion3.csv")
write.csv(confusion4, "C:/Users/yz22/Desktop/PHD R/EEG/confusion4.csv")
write.csv(confusion5, "C:/Users/yz22/Desktop/PHD R/EEG/confusion5.csv")
write.csv(confusion6, "C:/Users/yz22/Desktop/PHD R/EEG/confusion6.csv")
write.csv(confusion7, "C:/Users/yz22/Desktop/PHD R/EEG/confusion7.csv")
write.csv(confusion8, "C:/Users/yz22/Desktop/PHD R/EEG/confusion8.csv")
write.csv(confusion9, "C:/Users/yz22/Desktop/PHD R/EEG/confusion9.csv")
write.csv(confusion10, "C:/Users/yz22/Desktop/PHD R/EEG/confusion10.csv")

write.csv(confusionET1, "C:/Users/yz22/Desktop/PHD R/EEG/confusionET1.csv")
write.csv(confusionET2, "C:/Users/yz22/Desktop/PHD R/EEG/confusionET2.csv")
write.csv(confusionET3, "C:/Users/yz22/Desktop/PHD R/EEG/confusionET3.csv")
write.csv(confusionET4, "C:/Users/yz22/Desktop/PHD R/EEG/confusionET4.csv")
write.csv(confusionET5, "C:/Users/yz22/Desktop/PHD R/EEG/confusionET5.csv")
write.csv(confusionET6, "C:/Users/yz22/Desktop/PHD R/EEG/confusionET6.csv")
write.csv(confusionET7, "C:/Users/yz22/Desktop/PHD R/EEG/confusionET7.csv")
write.csv(confusionET8, "C:/Users/yz22/Desktop/PHD R/EEG/confusionET8.csv")
write.csv(confusionET9, "C:/Users/yz22/Desktop/PHD R/EEG/confusionET9.csv")
write.csv(confusionET10, "C:/Users/yz22/Desktop/PHD R/EEG/confusionET10.csv")
write.csv(confusionET11, "C:/Users/yz22/Desktop/PHD R/EEG/confusionET11.csv")
```

```
write.csv(confusionXgb1, "C:/Users/yz22/Desktop/PHD R/EEG/confusionXgb1.csv")
write.csv(confusionXgb2, "C:/Users/yz22/Desktop/PHD R/EEG/confusionXgb2.csv")
write.csv(confusionXgb3, "C:/Users/yz22/Desktop/PHD R/EEG/confusionXgb3.csv")
write.csv(confusionXgb4, "C:/Users/yz22/Desktop/PHD R/EEG/confusionXgb4.csv")
write.csv(confusionXgb5, "C:/Users/yz22/Desktop/PHD R/EEG/confusionXgb5.csv")
write.csv(confusionXgb6, "C:/Users/yz22/Desktop/PHD R/EEG/confusionXgb6.csv")
write.csv(confusionXgb7, "C:/Users/yz22/Desktop/PHD R/EEG/confusionXgb7.csv")
write.csv(confusionXgb8, "C:/Users/yz22/Desktop/PHD R/EEG/confusionXgb8.csv")
write.csv(confusionXgb9, "C:/Users/yz22/Desktop/PHD R/EEG/confusionXgb9.csv")
write.csv(confusionXgb10, "C:/Users/yz22/Desktop/PHD R/EEG/confusionXgb10.csv")

#####################################
##Multiple Correspondence Analysis##
#####################################
library(ca)
mca <- mjca(results[,2:32])
summary(mca, lambda = "Burt")
print(mca)

#standard coordinates of rows, row factor scores
mca$rowcoord[1:10,]
#Principla coordinates of rows, row factor scores, defined in the dissertation
mca$rowpcoord[1:10,]

plot.mjca(mca)
# add principle factor scores to the model results, create table resultMca, these data has two
factors
resultMca <- results
resultMca$PFS1 = mca$rowpcoord[,1]
resultMca$PFS2 = mca$rowpcoord[,2]

resultMca$FS1 = mca$rowcoord[,1]
resultMca$FS2 = mca$rowcoord[,2]


#############################################################
## CramÃ©r's V correlation coefficient of 30 base models##
#############################################################
library(vcd)

cramerx <- results[,c(-1)]

catcor <- function(x, type=c("cramer")) {
  require(vcd)
  nc <- ncol(x)
  v <- expand.grid(1:nc, 1:nc)
  type <- match.arg(type)
  res <- matrix(mapply(function(i1, i2) assocstats(table(x[,i1],
                                        x[,i2]))[[type]], v[,1], v[,2]), nc, nc)
  rownames(res) <- colnames(res) <- colnames(x)
```

```
  res
}

cramerxresult <- catcor(cramerx, type="cramer")

write.csv(cramerxresult, "C:/Users/yz22/Desktop/PHD R/EEG/cramerxresult.csv")


#######Second Stage: Experiment Design ###########

      ##########################################
      ## Experiment Design one #################
      ##########################################
      ## use xgboost to combine the base model##
      ##########################################
      ##############################################################
      install.packages("drat", repos="https://cran.rstudio.com")
      drat:::addRepo("dmlc")
      install.packages("xgboost", repos="http://dmlc.ml/drat/", type = "source")

      require(xgboost)
      set.seed(92754)

      #format data to fit Xgboost#
      #Xgboost requires all inputs are numberic

      resultsXgb=results

      #binarize all factors in the data set
      library(caret)
      dmy <- dummyVars(" ~ .", data=resultsXgb)
      resultsXgbdmy <- data.frame(predict(dmy, newdata=resultsXgb))

      dim(resultsXgbdmy)
      names(resultsXgbdmy)

      ##########################################################################

      #prepared a varialbe of target, and a matrix for predictor#
      outcomeName <-c('y')

      predictorsXgb <- names(resultsXgbdmy)[!names(resultsXgbdmy)%in% outcomeName]

      #set up parameters for Xgboost#
      param <- list("objective" = "binary:logistic",
                    "eval_metric" = "logloss",
                    "eta" =0.005, "max.depth" = 2)

      library(xgboost)

      #For variable importance
```

```
library(DiagrammeR)
library(Ckmeans.1d.dp)

#run 10 fold cross validation and choose the best round
bst.cv <- xgb.cv(param=param, data = as.matrix(resultsXgbdmy[,predictorsXgb]),
            label = resultsXgbdmy[,outcomeName], nfold=10, nround = 2000)
plot(log(bst.cv$test.logloss.mean),type ="l")

################################
#Xgboost Final Combined Model##
################################

bstComb <- xgboost(data = as.matrix(resultsXgbdmy[,predictorsXgb]),
              label = resultsXgbdmy[,outcomeName], max.depth = 2, eta = 0.005, nround
= 1540,
              nthread = 2, objective ="binary:logistic")
gc()

#make prediction#
predictionsXgb <- predict(bstComb, as.matrix(resultsXgbdmy[,predictorsXgb]),
outputmargin= FALSE) #outputmargin has to be FALSE to produce probability
#predictions[1:10]
print(-mean(log(predictionsXgb)*resultsXgbdmy$y+log(1-predictionsXgb)*(1-
resultsXgbdmy$y)))

# Get the variable importance
importance_matrix <- xgb.importance(predictorsXgb, model = bstComb)
xgb.plot.importance(importance_matrix[1:10])


#convert to categorical target and caculate the accuracy
yPredCatXgb <- ifelse(predictionsXgb <= 0.5,0,1)
yPredCatXgb[1:10]

confusionED1Xgb=table(yPredCatXgb,resultsXgbdmy$y)
accuracyXgb = sum(diag(confusionED1Xgb))/sum(confusionED1Xgb)
cat("xgboost Combine Base Model Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyXgb))
confusionED1Xgb
write.csv(confusionED1Xgb, "C:/Users/yz22/Desktop/PHD
R/EEG/confusionED1Xgb.csv")

#Create a new table to save the ensemble result
resultsFinal = data.frame(y = testX$y)
#save the ensemble result to table
resultsFinal$xgb.allBase = yPredCatXgb


#################################
##Random Forest Combined Model##
#################################
```

```
library(randomForest)

#create fisrt random forest#
modelRF = randomForest(resultsXgbdmy[,-1], as.factor(resultsXgbdmy[,1]),
                    xtest=resultsXgbdmy[,-1], ntree=50, set.seed(56382, kind=NULL,
normal.kind=NULL))
predictedRF = modelRF$test$predicted
resultsFinal$RF.allBase = predictedRF
confusionED1RF = table(predictedRF, resultsXgbdmy$y)
accuracyRF = sum(diag(confusionED1RF))/sum(confusionED1RF)
cat("FOREST Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyRF))
confusionED1RF
write.csv(confusionED1RF, "C:/Users/yz22/Desktop/PHD
R/EEG/confusionED1RF.csv")


##################################
## Majority Voting ##
##################################
require(functional)
confusionED1MV <- apply(results[,-1,drop=FALSE], 1, Compose(table,
                                        function(i) i==max(i),
                                        which,
                                        names,
                                        function(i) paste0(i, collapse='/')
)
)

resultsFinal$MV.allBase = confusionED1MV

confusionED1MV = table(confusionED1MV, resultsXgbdmy$y)
accuracyMV = sum(diag(confusionED1MV))/sum(confusionED1MV)
cat("Majority Voting Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyMV))
confusionED1MV
write.csv(confusionED1MV, "C:/Users/yz22/Desktop/PHD
R/EEG/confusionED1MV.csv")


##################################
## full logistic Regression ##
##################################
BLR <-
glm(y~as.factor(xgb1)+as.factor(xgb2)+as.factor(xgb3)+as.factor(xgb4)+as.factor(xgb5)

+as.factor(xgb6)+as.factor(xgb7)+as.factor(xgb8)+as.factor(xgb9)+as.factor(xgb10)

+as.factor(FOREST1)+as.factor(FOREST2)+as.factor(FOREST3)+as.factor(FOREST4)+as.f
actor(FOREST5)
```

```
+as.factor(FOREST6)+as.factor(FOREST7)+as.factor(FOREST8)+as.factor(FOREST9)+as.f
actor(FOREST10)
            +as.factor(ET1)+as.factor(ET2)+as.factor(ET3)+as.factor(ET4)+as.factor(ET5)

+as.factor(ET6)+as.factor(ET7)+as.factor(ET8)+as.factor(ET9)+as.factor(ET10)+as.factor(E
T11),
            family="binomial", data=resultsXgb)

      ##make prediction##
      Predictglm<-predict(BLR,resultsXgb[,-1],type="response")

      Predictglmcat <- ifelse(Predictglm <= 0.5,0,1)
      Predictglmcat[1:10]

      ####write to the final result table####
      resultsFinal$FullLR.allBase = Predictglmcat

      confusionED1LR=table(Predictglmcat,resultsXgb$y)
      accuracyLR = sum(diag(confusionED1LR))/sum(confusionED1LR)
      cat("Backward Logistic Regression Combine Base Model Results:\n")
      cat(sprintf("Accuracy %3.4f\n", accuracyLR))
      confusionED1LR
      write.csv(confusionED1LR, "C:/Users/yz22/Desktop/PHD
R/EEG/confusionED1LR.csv")


      ##########################################
      ## Experiment Design Two #################
      ##############################################################
      ## use xgboost to combine the base model and MCA factors##
      ##############################################################
      ##############################################################
      install.packages("drat", repos="https://cran.rstudio.com")
      drat:::addRepo("dmlc")
      install.packages("xgboost", repos="http://dmlc.ml/drat/", type = "source")

      require(xgboost)
      set.seed(73529)

      #format data to fit Xgboost#
      #Xgboost requires all inputs are numberic

      resultsMcaXgb=resultMca

      #binarize all factors in the data set
      library(caret)
      dmy <- dummyVars(" ~ .", data=resultsMcaXgb)
      resultsMcaXgbdmy <- data.frame(predict(dmy, newdata=resultsMcaXgb))

      dim(resultsMcaXgbdmy)
```

```
names(resultsMcaXgbdmy)

##########################################################################
#prepared a varialbe of target, and a matrix for predictor#
outcomeName <-c('y')

predictorsMcaXgb <- names(resultsMcaXgbdmy)[!names(resultsMcaXgbdmy)%in%
outcomeName]

#set up parameters for Xgboost#
param <- list("objective" = "binary:logistic",
              "eval_metric" = "logloss",
              "eta" =0.005, "max.depth" = 2)

library(xgboost)

#For variable importance
library(DiagrammeR)
library(Ckmeans.1d.dp)

#run 10 fold cross validation and choose the best round
bst.cv <- xgb.cv(param=param, data =
as.matrix(resultsMcaXgbdmy[,predictorsMcaXgb]),
                 label = resultsMcaXgbdmy[,outcomeName], nfold=10, nround = 2000)
plot(log(bst.cv$test.logloss.mean),type ="l")

################################
#Xgboost Final Combined Model##
################################

bstComb <- xgboost(data = as.matrix(resultsMcaXgbdmy[,predictorsMcaXgb]),
                   label = resultsMcaXgbdmy[,outcomeName], max.depth = 2, eta = 0.005,
nround = 2200,
                   nthread = 2, objective ="binary:logistic")
gc()
#make prediction#
predictionsMcaXgb <- predict(bstComb,
as.matrix(resultsMcaXgbdmy[,predictorsMcaXgb]), outputmargin= FALSE) #outputmargin
has to be FALSE to produce probability
#predictions[1:10]
print(-mean(log(predictionsMcaXgb)*resultsMcaXgbdmy$y+log(1-
predictionsMcaXgb)*(1-resultsMcaXgbdmy$y)))

# Get the variable importance
importance_matrix <- xgb.importance(predictorsMcaXgb, model = bstComb)
xgb.plot.importance(importance_matrix[1:10])

#convert to categorical target and caculate the accuracy
yPredCatMcaXgb <- ifelse(predictionsMcaXgb <= 0.5,0,1)
yPredCatMcaXgb[1:10]
```

```
confusionED2Xgb=table(yPredCatMcaXgb,resultsMcaXgbdmy$y)
accuracyMcaXgb = sum(diag(confusionED2Xgb))/sum(confusionED2Xgb)
cat("xgboost Combine Base Model Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyMcaXgb))
confusionED2Xgb
write.csv(confusionED2Xgb, "C:/Users/yz22/Desktop/PHD
R/EEG/confusionED2Xgb.csv")

#save the ensemble result to table, accuracy is 0.9544
resultsFinal$xgb.allBaseMCA = yPredCatMcaXgb

##################################
##Random Forest Combined Model##
##################################
library(randomForest)

#create fisrt random forest#
modelRF = randomForest(resultsMcaXgbdmy[,-1], as.factor(resultsMcaXgbdmy[,1]),
                       xtest=resultsMcaXgbdmy[,-1], ntree=50, set.seed(56382, kind=NULL,
normal.kind=NULL))
predictedRF = modelRF$test$predicted
resultsFinal$RF.allBaseMca = predictedRF
confusionED2RF = table(predictedRF, resultsXgbdmy$y)
accuracyRF = sum(diag(confusionED2RF))/sum(confusionED2RF)
cat("FOREST Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyRF))
confusionED2RF
write.csv(confusionED2RF, "C:/Users/yz22/Desktop/PHD
R/EEG/confusionED2RF.csv")

#################################
## full logistic Regression ##
#################################
BLRMca <-
glm(y~as.factor(xgb1)+as.factor(xgb2)+as.factor(xgb3)+as.factor(xgb4)+as.factor(xgb5)

+as.factor(xgb6)+as.factor(xgb7)+as.factor(xgb8)+as.factor(xgb9)+as.factor(xgb10)

+as.factor(FOREST1)+as.factor(FOREST2)+as.factor(FOREST3)+as.factor(FOREST4)+as.f
actor(FOREST5)

+as.factor(FOREST6)+as.factor(FOREST7)+as.factor(FOREST8)+as.factor(FOREST9)+as.f
actor(FOREST10)
            +as.factor(ET1)+as.factor(ET2)+as.factor(ET3)+as.factor(ET4)+as.factor(ET5)

+as.factor(ET6)+as.factor(ET7)+as.factor(ET8)+as.factor(ET9)+as.factor(ET10)+as.factor(E
T11)
            +PFS1+PFS2+FS1+FS2,
            family="binomial", data=resultsMcaXgb)

##make prediction##
```

```
Predictglm<-predict(BLRMca,resultsMcaXgb[,-1],type="response")

Predictglmcat <- ifelse(Predictglm <= 0.5,0,1)
Predictglmcat[1:10]

####write to the final result table####
resultsFinal$FullLR.allBase.Mca = Predictglmcat

confusionED2LR=table(Predictglmcat,resultsXgb$y)
accuracyLR = sum(diag(confusionED2LR))/sum(confusionED2LR)
cat("Backward Logistic Regression Combine Base Model Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyLR))
confusionED2LR
write.csv(confusionED2LR, "C:/Users/yz22/Desktop/PHD
R/EEG/confusionED2LR.csv")


##########################################
## Experiment Design Three #################
##########################################
## use xgboost to combine the ET and Xgb base model##
##########################################
#############################################################
install.packages("drat", repos="https://cran.rstudio.com")
drat:::addRepo("dmlc")
install.packages("xgboost", repos="http://dmlc.ml/drat/", type = "source")

require(xgboost)
set.seed(84620)

#format data to fit Xgboost#
#Xgboost requires all inputs are numberic

resultsXgbEX=results[-c(23:32)] ### only keep Xgb abd ET base model###

#binarize all factors in the data set
library(caret)
dmy <- dummyVars(" ~ .", data=resultsXgbEX)
resultsXgbEXdmy <- data.frame(predict(dmy, newdata=resultsXgbEX))

dim(resultsXgbdmy)
names(resultsXgbdmy)

#######################################################################
#prepared a varialbe of target, and a matrix for predictor#

outcomeName <-c('y')

predictorsXgb <- names(resultsXgbEXdmy)[!names(resultsXgbEXdmy)%in%
outcomeName]
```

```
#set up parameters for Xgboost#
param <- list("objective" = "binary:logistic",
              "eval_metric" = "logloss",
              "eta" =0.005, "max.depth" = 2)

library(xgboost)

#For variable importance
library(DiagrammeR)
library(Ckmeans.1d.dp)

#run 10 fold cross validation and choose the best round
bst.cv <- xgb.cv(param=param, data = as.matrix(resultsXgbEXdmy[,predictorsXgb]),
                 label = resultsXgbEXdmy[,outcomeName], nfold=10, nround = 2500)
plot(log(bst.cv$test.logloss.mean),type ="l")

################################
#Xgboost Final Combined Model##
################################

bstComb <- xgboost(data = as.matrix(resultsXgbEXdmy[,predictorsXgb]),
                   label = resultsXgbdmy[,outcomeName], max.depth = 2, eta = 0.005, nround
= 2005,
                   nthread = 2, objective ="binary:logistic")
gc()
#make prediction#
predictionsXgb <- predict(bstComb, as.matrix(resultsXgbEXdmy[,predictorsXgb]),
outputmargin= FALSE) #outputmargin has to be FALSE to produce probability
#predictions[1:10]
print(-mean(log(predictionsXgb)*resultsXgbEXdmy$y+log(1-predictionsXgb)*(1-
resultsXgbEXdmy$y)))

# Get the variable importance
importance_matrix <- xgb.importance(predictorsXgb, model = bstComb)
xgb.plot.importance(importance_matrix[1:10])

#convert to categorical target and caculate the accuracy
yPredCatXgb <- ifelse(predictionsXgb <= 0.5,0,1)
yPredCatXgb[1:10]

confusionED3Xgb=table(yPredCatXgb,resultsXgbEXdmy$y)
accuracyXgb = sum(diag(confusionED3Xgb))/sum(confusionED3Xgb)
cat("xgboost Combine Base Model Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyXgb))
confusionED3Xgb
write.csv(confusionED3Xgb, "C:/Users/yz22/Desktop/PHD
R/EEG/confusionED3Xgb.csv")

#save the ensemble result to table

resultsFinal$xgb.ETXgbBase = yPredCatXgb
```

```
#################################
##Random Forest Combined Model##
#################################
library(randomForest)

#create fisrt random forest#
modelRF = randomForest(resultsXgbEXdmy[,-1], as.factor(resultsXgbEXdmy[,1]),
                    xtest=resultsXgbEXdmy[,-1], ntree=50, set.seed(17395, kind=NULL,
normal.kind=NULL))
predictedRF = modelRF$test$predicted
resultsFinal$RF.ETXgbBase = predictedRF
confusionED3RF = table(predictedRF, resultsXgbEXdmy$y)
accuracyRF = sum(diag(confusionED3RF))/sum(confusionED3RF)
cat("FOREST Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyRF))
confusionED3RF
write.csv(confusionED3RF, "C:/Users/yz22/Desktop/PHD
R/EEG/confusionED3RF.csv")

################################
## Majority Voting ##
################################

require(functional)
confusionED3MV <- apply(resultsXgbEXdmy[,-1,drop=FALSE], 1, Compose(table,
                                        function(i) i==max(i),
                                        which,
                                        names,
                                        function(i) paste0(i, collapse='/')
)
)

resultsFinal$MV.ETXgbBase = confusionED3MV

confusionED3MV = table(confusionED3MV, resultsXgbEXdmy$y)
accuracyMV = sum(diag(confusionED3MV))/sum(confusionED3MV)
cat("Majority Voting Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyMV))
confusionED3MV
write.csv(confusionED3MV, "C:/Users/yz22/Desktop/PHD
R/EEG/confusionED3MV.csv")


##################################
## Backward logistic Regression ##
##################################
BLR <-
glm(y~as.factor(xgb1)+as.factor(xgb2)+as.factor(xgb3)+as.factor(xgb4)+as.factor(xgb5)

+as.factor(xgb6)+as.factor(xgb7)+as.factor(xgb8)+as.factor(xgb9)+as.factor(xgb10)
```

```
+as.factor(FOREST1)+as.factor(FOREST2)+as.factor(FOREST3)+as.factor(FOREST4)+as.f
actor(FOREST5)

+as.factor(FOREST6)+as.factor(FOREST7)+as.factor(FOREST8)+as.factor(FOREST9)+as.f
actor(FOREST10)
            +as.factor(ET1)+as.factor(ET2)+as.factor(ET3)+as.factor(ET4)+as.factor(ET5)

+as.factor(ET6)+as.factor(ET7)+as.factor(ET8)+as.factor(ET9)+as.factor(ET10)+as.factor(E
T11),
            family="binomial", data=resultsXgb)
      ##backward selection##
      BLRBack<-step(BLR,direction="backward")

      summary(BLRBack)

      ##make prediction##
      Predictglm<-predict(BLRBack,resultsXgb[,-1],type="response")

      Predictglmcat <- ifelse(Predictglm <= 0.5,0,1)
      Predictglmcat[1:10]

      ####write to the final result table####
      resultsFinal$LR.allBase = Predictglmcat

      confusionED3LR=table(Predictglmcat,resultsXgbEXdmy$y)
      accuracyLR = sum(diag(confusionED3LR))/sum(confusionED3LR)
      cat("Backward Logistic Regression Combine Base Model Results:\n")
      cat(sprintf("Accuracy %3.4f\n", accuracyLR))
      confusionED3LR
      write.csv(confusionED3LR, "C:/Users/yz22/Desktop/PHD
R/EEG/confusionED3LR.csv")


      ##########################################
      ## Experiment Design Four and Five#################
      ########################################################
      ## use xgboost to combine the ET and Xgb base model and MCA##
      ########################################################

      install.packages("drat", repos="https://cran.rstudio.com")
      drat:::addRepo("dmlc")
      install.packages("xgboost", repos="http://dmlc.ml/drat/", type = "source")

      require(xgboost)
      set.seed(93746)

      #format data to fit Xgboost#
      #Xgboost requires all inputs are numberic

      resultsXgbEXMca=resultMca[-c(23:32)] ### only keep Xgb abd ET base model###
```

```
#binarize all factors in the data set
library(caret)
dmy <- dummyVars(" ~ .", data=resultsXgbEXMca)
resultsXgbEXMcadmy <- data.frame(predict(dmy, newdata=resultsXgbEXMca))

dim(resultsXgbEXMcadmy)
names(resultsXgbEXMcadmy)

##########################################################################
#prepared a varialbe of target, and a matrix for predictor#
outcomeName <-c('y')

predictorsXgb <- names(resultsXgbEXMcadmy)[!names(resultsXgbEXMcadmy)%in%
outcomeName]

#set up parameters for Xgboost#
param <- list("objective" = "binary:logistic",
              "eval_metric" = "logloss",
              "eta" =0.005, "max.depth" = 2)

library(xgboost)

#For variable importance
library(DiagrammeR)
library(Ckmeans.1d.dp)

#run 10 fold cross validation and choose the best round;round 80 to 120 all have the
relative similar accuracy
bst.cv <- xgb.cv(param=param, data =
as.matrix(resultsXgbEXMcadmy[,predictorsXgb]),
                 label = resultsXgbEXMcadmy[,outcomeName], nfold=10, nround = 2500)
plot(log(bst.cv$test.logloss.mean),type ="l")

#################################
#Xgboost Final Combined Model##
#################################

bstComb <- xgboost(data = as.matrix(resultsXgbEXMcadmy[,predictorsXgb]),
                   label = resultsXgbEXMcadmy[,outcomeName], max.depth = 2, eta = 0.005,
nround = 2500,
                   nthread = 2, objective ="binary:logistic")
gc()
#make prediction#
predictionsXgb <- predict(bstComb, as.matrix(resultsXgbEXMcadmy[,predictorsXgb]),
outputmargin= FALSE) #outputmargin has to be FALSE to produce probability
#predictions[1:10]
print(-mean(log(predictionsXgb)*resultsXgbEXMcadmy$y+log(1-predictionsXgb)*(1-
resultsXgbEXMcadmy$y)))

# Get the variable importance
```

```
importance_matrix <- xgb.importance(predictorsXgb, model = bstComb)
xgb.plot.importance(importance_matrix[1:10])


#convert to categorical target and caculate the accuracy
yPredCatXgb <- ifelse(predictionsXgb <= 0.5,0,1)
yPredCatXgb[1:10]

confusionED4Xgb=table(yPredCatXgb,resultsXgbEXMcadmy$y)
accuracyXgb = sum(diag(confusionED4Xgb))/sum(confusionED4Xgb)
cat("xgboost Combine Base Model Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyXgb))
confusionED4Xgb
write.csv(confusionED4Xgb, "C:/Users/yz22/Desktop/PHD
R/EEG/confusionED4Xgb.csv")

#save the ensemble result to table
resultsFinal$xgb.ETXgbBase.Mca = yPredCatXgb


################################
##Random Forest Combined Model##
################################
library(randomForest)

#create fisrt random forest#
modelRF = randomForest(resultsXgbEXMcadmy[,-1],
as.factor(resultsXgbEXMcadmy[,1]),
                       xtest=resultsXgbEXMcadmy[,-1], ntree=50, set.seed(86527, kind=NULL,
normal.kind=NULL))
predictedRF = modelRF$test$predicted
resultsFinal$RF.ETXgbBase.Mca = predictedRF
confusionED4RF = table(predictedRF, resultsXgbEXMcadmy$y)
accuracyRF = sum(diag(confusionED4RF))/sum(confusionED4RF)
cat("FOREST Results:\n")
cat(sprintf("Accuracy %3.4f\n", accuracyRF))
confusionED4RF
write.csv(confusionED4RF, "C:/Users/yz22/Desktop/PHD
R/EEG/confusionED4RF.csv")


##################################
## Backward logistic Regression ##
##################################
BLRMca <-
glm(y~as.factor(xgb1)+as.factor(xgb2)+as.factor(xgb3)+as.factor(xgb4)+as.factor(xgb5)

+as.factor(xgb6)+as.factor(xgb7)+as.factor(xgb8)+as.factor(xgb9)+as.factor(xgb10)

+as.factor(FOREST1)+as.factor(FOREST2)+as.factor(FOREST3)+as.factor(FOREST4)+as.f
actor(FOREST5)
```

```
+as.factor(FOREST6)+as.factor(FOREST7)+as.factor(FOREST8)+as.factor(FOREST9)+as.f
actor(FOREST10)
                +as.factor(ET1)+as.factor(ET2)+as.factor(ET3)+as.factor(ET4)+as.factor(ET5)

+as.factor(ET6)+as.factor(ET7)+as.factor(ET8)+as.factor(ET9)+as.factor(ET10)+as.factor(E
T11)
                +PFS1+PFS2+FS1+FS2,
                family="binomial", data=resultsMcaXgb)
        ##backward selection##
        BLRBackMca<-step(BLRMca,direction="backward")

        summary(BLRBackMca)

        ##make prediction##
        Predictglm<-predict(BLRBackMca,resultsXgb[,-1],type="response")

        Predictglmcat <- ifelse(Predictglm <= 0.5,0,1)
        Predictglmcat[1:10]

        ####write to the final result table####
        resultsFinal$LR.allBaseMca = Predictglmcat

        confusionED4LR=table(Predictglmcat,resultsMcaXgb$y)
        accuracyLR = sum(diag(confusionED4LR))/sum(confusionED4LR)
        cat("Backward Logistic Regression Combine Base Model Results:\n")
        cat(sprintf("Accuracy %3.4f\n", accuracyLR))
        confusionED4LR
        write.csv(confusionED4LR, "C:/Users/yz22/Desktop/PHD
R/EEG/confusionED4LR.csv")
```

Appendix C: Cramér's V Correlation Coefficient of Adult Data Set

	xgb1	xgb2	xgb3	xgb4	xgb5	xgb6	xgb7	xgb8	xgb9	xgb10
xgb1	1									
xgb2	0.9128	1								
xgb3	0.8912	0.8989	1							
xgb4	0.9022	0.9103	0.9107	1						
xgb5	0.9130	0.9181	0.9131	0.9219	1					
xgb6	0.9071	0.9183	0.9133	0.9190	0.9341	1				
xgb7	0.9159	0.9203	0.9138	0.9233	0.9411	0.9509	1			
xgb8	0.9126	0.9162	0.9073	0.9142	0.9367	0.9546	0.9551	1		
xgb9	0.8979	0.9091	0.8921	0.9029	0.9273	0.9410	0.9380	0.9586	1	
xgb10	0.8813	0.8930	0.8762	0.8836	0.9065	0.9252	0.9203	0.9385	0.9626	1
ET1	0.7099	0.7152	0.7070	0.7097	0.7218	0.7201	0.7177	0.7227	0.7285	0.7241
ET2	0.7119	0.7134	0.7108	0.7101	0.7223	0.7221	0.7231	0.7273	0.7305	0.7271
ET3	0.7110	0.7166	0.7099	0.7115	0.7240	0.7215	0.7226	0.7264	0.7314	0.7289
ET4	0.7109	0.7166	0.7117	0.7133	0.7262	0.7271	0.7236	0.7293	0.7321	0.7296
ET5	0.7116	0.7169	0.7117	0.7121	0.7250	0.7240	0.7236	0.7278	0.7332	0.7299
ET6	0.7146	0.7196	0.7129	0.7137	0.7266	0.7260	0.7251	0.7293	0.7348	0.7307
ET7	0.7129	0.7197	0.7148	0.7126	0.7263	0.7249	0.7234	0.7287	0.7337	0.7304
ET8	0.7135	0.7191	0.7139	0.7140	0.7276	0.7259	0.7258	0.7300	0.7343	0.7307
ET9	0.7134	0.7194	0.7120	0.7131	0.7253	0.7243	0.7239	0.7284	0.7339	0.7298
ET10	0.7140	0.7196	0.7129	0.7137	0.7278	0.7257	0.7256	0.7313	0.7360	0.7323
ET11	0.7133	0.7190	0.7153	0.7157	0.7286	0.7273	0.7276	0.7314	0.7365	0.7324
RF1	0.8852	0.8953	0.8905	0.8952	0.9057	0.9185	0.9144	0.9255	0.9241	0.9135
RF2	0.8281	0.8397	0.8271	0.8318	0.8529	0.8587	0.8558	0.8652	0.8781	0.8857
RF3	0.8265	0.8393	0.8262	0.8309	0.8537	0.8555	0.8554	0.8640	0.8741	0.8813
RF4	0.8293	0.8409	0.8287	0.8284	0.8541	0.8564	0.8558	0.8645	0.8797	0.8854
RF5	0.8254	0.8378	0.8255	0.8283	0.8522	0.8551	0.8546	0.8636	0.8738	0.8809
RF6	0.8295	0.8404	0.8297	0.8332	0.8563	0.8593	0.8580	0.8678	0.8779	0.8844
RF7	0.8267	0.8375	0.8265	0.8288	0.8535	0.8569	0.8544	0.8642	0.8747	0.8815
RF8	0.8288	0.8419	0.8293	0.8309	0.8552	0.8593	0.8572	0.8671	0.8799	0.8860
RF9	0.8254	0.8370	0.8232	0.8264	0.8526	0.8532	0.8523	0.8617	0.8749	0.8822
RF10	0.8282	0.8410	0.8287	0.8315	0.8546	0.8584	0.8579	0.8681	0.8786	0.8854

	ET1	ET2	ET3	ET4	ET5	ET6	ET7	ET8	ET9	ET10	ET11
ET1	1										
ET2	0.9324	1									
ET3	0.9389	0.9511	1								
ET4	0.9339	0.9576	0.9622	1							
ET5	0.9394	0.9557	0.9611	0.9627	1						
ET6	0.9387	0.9579	0.9629	0.9631	0.9723	1					
ET7	0.9434	0.9579	0.9673	0.9668	0.9701	0.9712	1				
ET8	0.9442	0.9598	0.9666	0.9686	0.9675	0.9723	0.9734	1			
ET9	0.9429	0.9596	0.9661	0.9677	0.9718	0.9714	0.9769	0.9766	1		
ET10	0.9411	0.9596	0.9646	0.9670	0.9695	0.9732	0.9762	0.9780	0.9790	1	
ET11	0.9446	0.9612	0.9644	0.9697	0.9716	0.9723	0.9741	0.9767	0.9751	0.9769	1
RF1	0.7244	0.7252	0.7274	0.7315	0.7321	0.7296	0.7327	0.7318	0.7302	0.7323	0.7327
RF2	0.7777	0.7801	0.7826	0.7856	0.7848	0.7852	0.7865	0.7859	0.7851	0.7872	0.7869
RF3	0.7810	0.7834	0.7863	0.7874	0.7896	0.7889	0.7890	0.7869	0.7895	0.7902	0.7914
RF4	0.7834	0.7874	0.7887	0.7914	0.7913	0.7921	0.7923	0.7917	0.7924	0.7937	0.7942
RF5	0.7776	0.7835	0.7848	0.7886	0.7882	0.7875	0.7895	0.7878	0.7877	0.7891	0.7899
RF6	0.7821	0.7876	0.7904	0.7923	0.7922	0.7927	0.7932	0.7915	0.7937	0.7943	0.7959
RF7	0.7835	0.7868	0.7884	0.7903	0.7922	0.7922	0.7920	0.7906	0.7921	0.7942	0.7939
RF8	0.7791	0.7839	0.7856	0.7878	0.7889	0.7874	0.7891	0.7870	0.7892	0.7890	0.7910
RF9	0.7830	0.7878	0.7894	0.7909	0.7916	0.7921	0.7933	0.7905	0.7923	0.7925	0.7942
RF10	0.7812	0.7874	0.7880	0.7910	0.7906	0.7906	0.7919	0.7898	0.7916	0.7919	0.7931

	RF1	RF2	RF3	RF4	RF5	RF6	RF7	RF8	RF9	RF10
RF1	1									
RF2	0.8535	1								
RF3	0.8514	0.9547	1							
RF4	0.8536	0.9575	0.9633	1						
RF5	0.8546	0.9629	0.9659	0.9655	1					
RF6	0.8513	0.9619	0.9689	0.9665	0.9730	1				
RF7	0.8536	0.9659	0.9682	0.9720	0.9731	0.9733	1			
RF8	0.8550	0.9630	0.9691	0.9683	0.9749	0.9747	0.9767	1		
RF9	0.8519	0.9612	0.9697	0.9712	0.9727	0.9737	0.9757	0.9798	1	
RF10	0.8543	0.9626	0.9700	0.9704	0.9746	0.9752	0.9749	0.9758	0.9799	1

Appendix D: Cramér's V Correlation Coefficient of Credit Card Client Data Set

	xgb1	**xgb2**	**xgb3**	**xgb4**	**xgb5**	**xgb6**	**xgb7**	**xgb8**	**xgb9**	**xgb10**
xgb1	1									
xgb2	0.8214	1								
xgb3	0.7944	0.8906	1							
xgb4	0.7923	0.899	0.9349	1						
xgb5	0.8027	0.8912	0.9184	0.9324	1					
xgb6	0.7987	0.8914	0.9063	0.9316	0.93	1				
xgb7	0.7983	0.8909	0.9121	0.9343	0.9337	0.9522	1			
xgb8	0.8064	0.8994	0.9112	0.9345	0.9348	0.9342	0.9518	1		
xgb9	0.7932	0.8917	0.9045	0.933	0.9271	0.9466	0.9599	0.9514	1	
xgb10	0.7984	0.8927	0.9129	0.9394	0.9323	0.9498	0.9642	0.9546	0.9766	1
ET1	0.0021	0.0076	0.0013	0.0032	0.0043	0.0021	0.005	0.0007	0.0035	0.0025
ET2	0.004	0.0076	0.0013	0.0033	0.0053	0.0032	0.0061	0.0014	0.0046	0.0036
ET3	0.0024	0.0049	0.0041	0.0004	0.0025	0.0003	0.0042	0.0015	0.0027	0.0017
ET4	0.0046	0.0091	0.0008	0.0057	0.0077	0.0046	0.0085	0.0038	0.007	0.006
ET5	0.0046	0.0091	0.0008	0.0057	0.0077	0.0046	0.0085	0.0038	0.007	0.006
ET6	0.0046	0.0091	0.0008	0.0057	0.0077	0.0046	0.0085	0.0038	0.007	0.006
ET7	0.0046	0.0091	0.0008	0.0057	0.0077	0.0046	0.0085	0.0038	0.007	0.006
ET8	0.0034	0.0079	0.0021	0.0035	0.0045	0.0024	0.0063	0.0016	0.0048	0.0038
ET9	0.0034	0.0079	0.0021	0.0035	0.0045	0.0024	0.0063	0.0016	0.0048	0.0038
ET10	0.0034	0.0079	0.0021	0.0035	0.0045	0.0024	0.0063	0.0016	0.0048	0.0038
ET11	0.0034	0.0079	0.0021	0.0035	0.0045	0.0024	0.0063	0.0016	0.0048	0.0038
RF1	0.0084	0.0023	0.0116	0.0074	0.006	0.0087	0.0016	0.0092	0.0051	0.0061
RF2	0.0104	0.0023	0.0107	0.0055	0.0051	0.0058	0.0007	0.0063	0.0022	0.0032
RF3	0.0103	0.0032	0.0138	0.0098	0.0082	0.0101	0.0061	0.0095	0.0064	0.0075
RF4	0.0074	0.0013	0.0128	0.0087	0.0072	0.0059	0.0008	0.0084	0.0043	0.0053
RF5	0.0051	0.0031	0.0075	0.0044	0.0029	0.0036	0.0015	0.0041	0.002	0.0031
RF6	0.0095	0.0024	0.014	0.0089	0.0074	0.0071	0.0031	0.0086	0.0055	0.0066
RF7	0.0071	0	0.0116	0.0065	0.0049	0.0057	0.0017	0.0061	0.0041	0.0052
RF8	0.0062	0.0019	0.0107	0.0067	0.0051	0.0049	0.0002	0.0063	0.0022	0.0033
RF9	0.0077	0.0017	0.0112	0.0061	0.0056	0.0064	0.0023	0.0058	0.0038	0.0048
RF10	0.0058	0.0004	0.0121	0.007	0.0055	0.0062	0.0022	0.0067	0.0036	0.0047

	ET1	ET2	ET3	ET4	ET5	ET6	ET7	ET8	ET9	ET10	ET11
ET1	1										
ET2	0.8837	1									
ET3	0.8992	0.9175	1								
ET4	0.8965	0.9252	0.9362	1							
ET5	0.8965	0.9252	0.9362	1	1						
ET6	0.8965	0.9252	0.9362	1	1	1					
ET7	0.8965	0.9252	0.9362	1	1	1	1				
ET8	0.9042	0.9262	0.9419	0.9525	0.9525	0.9525	0.9525	1			
ET9	0.9042	0.9262	0.9419	0.9525	0.9525	0.9525	0.9525	1	1		
ET10	0.9042	0.9262	0.9419	0.9525	0.9525	0.9525	0.9525	1	1	1	
ET11	0.9042	0.9262	0.9419	0.9525	0.9525	0.9525	0.9525	1	1	1	1
RF1	0.8035	0.81	0.8154	0.8226	0.8226	0.8226	0.8226	0.8209	0.8209	0.8209	0.8209
RF2	0.8022	0.8137	0.8161	0.8303	0.8303	0.8303	0.8303	0.8197	0.8197	0.8197	0.8197
RF3	0.8143	0.8219	0.8292	0.8357	0.8357	0.8357	0.8357	0.8349	0.8349	0.8349	0.8349
RF4	0.8097	0.8182	0.8226	0.8358	0.8358	0.8358	0.8358	0.8291	0.8291	0.8291	0.8291
RF5	0.8082	0.8217	0.8261	0.8384	0.8384	0.8384	0.8384	0.8297	0.8297	0.8297	0.8297
RF6	0.8137	0.8223	0.8267	0.837	0.837	0.837	0.837	0.8343	0.8343	0.8343	0.8343
RF7	0.8161	0.8325	0.834	0.8463	0.8463	0.8463	0.8463	0.8396	0.8396	0.8396	0.8396
RF8	0.8123	0.8229	0.8282	0.8396	0.8396	0.8396	0.8396	0.8359	0.8359	0.8359	0.8359
RF9	0.8164	0.8309	0.8314	0.8467	0.8467	0.8467	0.8467	0.8419	0.8419	0.8419	0.8419
RF10	0.8191	0.8345	0.837	0.8482	0.8482	0.8482	0.8482	0.8425	0.8425	0.8425	0.8425

	RF1	RF2	RF3	RF4	RF5	RF6	RF7	RF8	RF9	RF10
RF1	1									
RF2	0.8839	1								
RF3	0.8805	0.9078	1							
RF4	0.8846	0.9118	0.9228	1						
RF5	0.8903	0.9197	0.9286	0.9357	1					
RF6	0.8839	0.9121	0.9251	0.9394	0.9391	1				
RF7	0.8883	0.9156	0.9337	0.9367	0.9426	0.9493	1			
RF8	0.8866	0.9149	0.9248	0.933	0.943	0.9467	0.9502	1		
RF9	0.8897	0.9191	0.9404	0.9392	0.9462	0.9406	0.9574	0.9465	1	
RF10	0.8942	0.9246	0.9408	0.9396	0.9516	0.9522	0.9577	0.952	0.9521	1

Appendix E: Cramér's V Correlation Coefficient of EEG Eye State Data Set

	xgb1	**xgb2**	**xgb3**	**xgb4**	**xgb5**	**xgb6**	**xgb7**	**xgb8**	**xgb9**	**xgb10**
xgb1	1									
xgb2	0.8755	1								
xgb3	0.8760	0.8886	1							
xgb4	0.8959	0.8968	0.8891	1						
xgb5	0.8877	0.9075	0.8980	0.9234	1					
xgb6	0.8909	0.8999	0.9066	0.9238	0.9274	1				
xgb7	0.8905	0.9031	0.8981	0.9144	0.9261	0.9328	1			
xgb8	0.8755	0.8908	0.8885	0.9058	0.9120	0.9251	0.9274	1		
xgb9	0.8565	0.8673	0.8686	0.8814	0.8785	0.8971	0.9076	0.9286	1	
xgb10	0.7905	0.8021	0.7980	0.8128	0.8088	0.8230	0.8337	0.8506	0.8841	1
ET1	0.8072	0.8180	0.8103	0.8268	0.8247	0.8234	0.8286	0.8211	0.8156	0.7795
ET2	0.8032	0.8167	0.8099	0.8326	0.8261	0.8302	0.8290	0.8227	0.8181	0.7851
ET3	0.8158	0.8302	0.8243	0.8417	0.8378	0.8428	0.8462	0.8380	0.8307	0.7894
ET4	0.8140	0.8338	0.8216	0.8426	0.8405	0.8420	0.8435	0.8390	0.8299	0.7887
ET5	0.8090	0.8252	0.8112	0.8394	0.8319	0.8361	0.8394	0.8321	0.8257	0.7880
ET6	0.8149	0.8284	0.8198	0.8444	0.8396	0.8429	0.8444	0.8371	0.8307	0.7895
ET7	0.8099	0.8261	0.8184	0.8403	0.8346	0.8379	0.8394	0.8312	0.8275	0.7879
ET8	0.8144	0.8270	0.8229	0.8439	0.8364	0.8406	0.8439	0.8357	0.8312	0.7890
ET9	0.8122	0.8293	0.8216	0.8435	0.8369	0.8428	0.8462	0.8380	0.8334	0.7940
ET10	0.8122	0.8275	0.8189	0.8407	0.8360	0.8392	0.8426	0.8353	0.8307	0.7886
ET11	0.8122	0.8275	0.8189	0.8407	0.8360	0.8392	0.8426	0.8353	0.8307	0.7886
RF1	0.8036	0.8089	0.8030	0.8204	0.8201	0.8270	0.8250	0.8212	0.8256	0.7959
RF2	0.8122	0.8284	0.8179	0.8390	0.8369	0.8383	0.8399	0.8388	0.8388	0.8064
RF3	0.8059	0.8230	0.8134	0.8336	0.8333	0.8384	0.8373	0.8334	0.8351	0.8008
RF4	0.8131	0.8248	0.8207	0.8408	0.8351	0.8402	0.8409	0.8343	0.8387	0.8017
RF5	0.8117	0.8252	0.8202	0.8358	0.8373	0.8397	0.8422	0.8366	0.8383	0.8024
RF6	0.8149	0.8284	0.8207	0.8381	0.8369	0.8438	0.8408	0.8352	0.8370	0.8001
RF7	0.8135	0.8270	0.8184	0.8367	0.8400	0.8424	0.8449	0.8366	0.8392	0.8050
RF8	0.8135	0.8279	0.8202	0.8394	0.8373	0.8415	0.8395	0.8366	0.8383	0.7996
RF9	0.8122	0.8184	0.8143	0.8335	0.8314	0.8383	0.8363	0.8334	0.8379	0.8010
RF10	0.8144	0.8234	0.8175	0.8331	0.8355	0.8388	0.8386	0.8348	0.8392	0.8024

	ET1	ET2	ET3	ET4	ET5	ET6	ET7	ET8	ET9	ET10	ET11
ET1	1										
ET2	0.9352	1									
ET3	0.9361	0.9547	1								
ET4	0.9425	0.9529	0.9611	1							
ET5	0.9447	0.9561	0.9606	0.9624	1						
ET6	0.9434	0.9565	0.9583	0.9629	0.9669	1					
ET7	0.9401	0.9570	0.9678	0.9615	0.9719	0.9633	1				
ET8	0.9438	0.9597	0.9669	0.9606	0.9710	0.9715	0.9665	1			
ET9	0.9442	0.9574	0.9683	0.9665	0.9724	0.9656	0.9714	0.9724	1		
ET10	0.9461	0.9565	0.9665	0.9692	0.9696	0.9665	0.9696	0.9715	0.9710	1	
ET11	0.9461	0.9565	0.9665	0.9692	0.9696	0.9665	0.9696	0.9715	0.9710	1	1
RF1	0.9082	0.9053	0.9070	0.9108	0.9174	0.9062	0.9110	0.9066	0.9134	0.9125	0.9125
RF2	0.9124	0.9212	0.9247	0.9293	0.9270	0.9248	0.9233	0.9252	0.9284	0.9302	0.9302
RF3	0.9177	0.9221	0.9211	0.9276	0.9306	0.9257	0.9242	0.9279	0.9311	0.9284	0.9284
RF4	0.9250	0.9266	0.9265	0.9366	0.9379	0.9338	0.9360	0.9352	0.9420	0.9366	0.9366
RF5	0.9237	0.9244	0.9333	0.9407	0.9356	0.9334	0.9347	0.9365	0.9406	0.9352	0.9352
RF6	0.9260	0.9312	0.9320	0.9393	0.9433	0.9329	0.9369	0.9370	0.9438	0.9429	0.9429
RF7	0.9228	0.9298	0.9324	0.9361	0.9356	0.9361	0.9337	0.9347	0.9379	0.9379	0.9379
RF8	0.9273	0.9289	0.9324	0.9407	0.9438	0.9334	0.9365	0.9374	0.9415	0.9397	0.9397
RF9	0.9242	0.9312	0.9302	0.9375	0.9388	0.9338	0.9333	0.9397	0.9411	0.9438	0.9438
RF10	0.9192	0.9307	0.9306	0.9389	0.9383	0.9316	0.9319	0.9347	0.9388	0.9370	0.9370

	RF1	RF2	RF3	RF4	RF5	RF6	RF7	RF8	RF9	RF10
RF1	1									
RF2	0.9187	1								
RF3	0.9277	0.9446	1							
RF4	0.9305	0.9446	0.9509	1						
RF5	0.9283	0.9469	0.9514	0.9605	1					
RF6	0.9369	0.9492	0.9564	0.9619	0.9560	1				
RF7	0.9328	0.9523	0.9532	0.9614	0.9600	0.9632	1			
RF8	0.9373	0.9542	0.9569	0.9632	0.9628	0.9660	0.9646	1		
RF9	0.9351	0.9519	0.9528	0.9619	0.9641	0.9673	0.9650	0.9714	1	
RF10	0.9355	0.9569	0.9578	0.9623	0.9655	0.9669	0.9700	0.9737	0.9750	1

References

Abdelazeem, S. (2008). A greedy approach for building classification cascades. *In Proceedings of the Seventh International Conference on Machine Learning and Applications, San Diego, CA, USA,* 115–120.

Abdi, H., & Valentin, D. (2007). In N. Salkind (Ed.), *Multiple Correspondence Analysis.* Encyclopedia of Measurement and Statistics. Thousand Oaks (CA): Sage.

Banfield, R. E., Hall, L. O., Bowyer, K. W., & Kegelmeyer, W. P. (2007). A comparison of decision tree ensemble creation techniques. *IEEE Trans. Pattern Anal. Mach. Intell, 29*(1), 173–180.

Bauer, E., & Kohavi, R. (1999). An empirical comparison of voting classification algorithms: bagging, boosting and variants. *Machine Learning 36*(1–2), 105–139.

Blake, C., & Merz, C. (1998). *UCI repository of machine learning database*. Technical report, Department of Information and Computer Science, University of California, Irvine, CA.

Bozdogan, H. (1987). Model Selection and Akaike's Information Criterion (AIC): The General Theory and Its Analytical Extensions. *Psychometrika, 52*, 345–370.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees.* Monterey, CA: Wadsworth & Brooks / Cole Advanced Books & Software.

Breiman, L. (1996). Bagging predictors. *Machine Learning, 24*(2), 123–140.

Breiman, L. (1996). Stacked regressions. *Machine Learning, 24*(1), 49–64.

Breiman, L. (1998). Arcing classifiers. *The Annals of Statistics, 26*(3), 801–849.

Breiman, L. (2001). Random forests. *Machine Learning, 45*(1), 5–32.

Breiman, L., Cutler, A, Liaw, A., & Wiener, M. (2015). Package 'randomForest'. Retrieved from https://cran.r-project.org/web/packages/randomForest/randomForest.pdf

Bryll, R., Gutierrez-Osuna, R., & Quek, F. K. (2003). Attribute bagging: improving accuracy of classiffer ensembles. *Pattern Recognition 36*(6), 1291–1302.

Caruana, R., & Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms. *In Proc. of the 23rd International Conference on Machine Learning*, 161–168.

Cevikalp, H., & Polikar, R. (2008). Local Classifier Weighting by Quadratic Programming. *IEEE Transactions on Neural Networks, 19*(10), 1832–1838.

Chen, T. (2014). *Introduction to Boosted Trees*. Retrieved from https://homes.cs.washington.edu/~tqchen/pdf/BoostedTree.pdf

Chen, T., He, T., & Benesty, M. (2016). *Package 'xgboost'*. Retrieved from https://cran.r-project.org/web/packages/xgboost/xgboost.pdf

Chi, D., Yeh, C., & Lai, M. (2011). A Hybrid Approach of DEA, Rough Set Theory and Random Forests for Credits. *International Journal of Innovative Computing, Information and Control, 7*(8), 4885-4897.

Cramér, H. (1946). *Mathematical Methods of Statistics*. Princeton: Princeton University Press.

David, D., Achim, Z., Kurt, H., Florian, G., & Michael, F. (2016) Package 'vcd'. Retrieved from https://cran.r-project.org/web/packages/vcd/vcd.pdf

Diaz-Uriarte, R., & Alvarez de Andres, S. (2006). Gene selection and classification of microarray data using random forest. *BMC Bioinformatics, 7*(3).

Dietterich, T. G. (2000). An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization. *Machine Learning, 40*(2), 139–157.

Dimililer, N., Varoglu, E., & Altincay, H. (2007). Vote-based classifier selection for biomedical NER using genetic algorithm. *In Proceedings of the 3rd Iberian Conference on Pattern Recognition and Image Analysis, Girona, Spain*, 202–209.

Dzeroski, S. & Zenko, B. (2004). Is combining classifiers with stacking better than selecting the best one? *Machine Learning 54*(3), 255–273.

Eddelbuettel, D., Boettiger, C., Gibb, S., Gillespie, C., Górecki, J., Jones, M., Leeper, T, Pav, S., & Schulz, J. (2016). Package 'drat'. Retrieved from https://cran.rstudio.com/web/packages/drat/drat.pdf

Freund, Y., & Shapire, R. E. (1996). Experiments with a new boosting algorithm. *Proceedings of the Thirteenth National Conference on Machine Learning*, 148–156.

Freund, Y, & Schapire R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *J Comput Syst Sci, 55*(1), 119–139.(

Freund, Y., & Schapire, R. E. (1999). A short introduction to boosting. *Japanese Society for Artificial Intelligence, 14*(5), 771–780.

Friedman, J. (2001). Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, 1189-1232.

Friedman, J., Hastie, T., & Tibshirani, R. (2000). Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics, 28*(2), 337–407.

Gan, Z. G., & Xiao, N. F. (2009). A new ensemble learning algorithm based on improved K-Means. *International Symposium on Intelligent Information Technology and Security Informatics, Moscow, Russia,* 8–11.

Geurts, P., Fillet, M., de Seny, D., Meuwis, M. A., Merville, M. P., & Wehenkel, L. (2005). Proteomic mass spectra classification using decision tree based ensemble methods. *Bioinformatics, 21*(14), 3138–3145.

Geurts, P., & Wehenkel, L. (2000). Investigation and reduction of discretization variance in decision tree induction. *Proceedings of the 11th European Conference on Machine Learning,* 162–170.

Geurts, P., & Wehenkel, L. (2005). Segment and combine approach for non-parametric time-series classification. *Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases,* 478–485.

Greenacre, M., & Blasius, J. (2006). *Multiple Correspondence Analysis and Related Methods.* London: Chapman & Hall/CRC.

Greenacre, M., Nenadic, O., & Friendly, M. (2016). *Package 'ca'*. Retrieved from https://cran.r-project.org/web/packages/ca/ca.pdf

Hu, X. (2001). Using rough sets theory and database operations to construct a good ensemble of classifiers for data mining applications. *In Proceedings of the 1st IEEE International Conference on Data Mining, San Jose, CA, USA*, 233–240.

Hocking, R. (1976) The Analysis and Selection of Variables in Linear Regression. *Biometrics, 32*(1), 1-49.

Iannone, R. (2016). *Package 'DiagrammeR'*. Retrieved from https://cran.r-project.org/web/packages/DiagrammeR/DiagrammeR.pdf

Jurek, A., Bi, Y., Wu, S., & Nugent, C. (2011). Classification by clusters analysis-an ensemble technique in a semi-supervised classification. *In 23rd IEEE International Conference on Tools with Artificial Intelligence, Boca Raton, FL, USA,* 876–878.

Jurek, A., Bi, Y., Wu, S., & Nugent, C. (2013). A survey of commonly used ensemble-based classification techniques. *The Knowledge Engineering*

Review, 29(5), 551-581.

Johnson, R., & Zhang, T. (2014). Learning Nonlinear Functions Using Regularized Greedy Forest. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 36*(5), 942–954.

Kittler, J., & Roli, F. (2001). Genetic algorithms for multi-classifier system configuration: a case study in character recognition. *In Proceedings of the 2nd International Workshop on Multiple Classifier System, Cambridge, UK,* 99–108.

Kohavi, R., & Wolpert, D. (1996). Bias plus variance decomposition for zero-one loss functions. *In Proceedings of the 13th International Conference on Machine Learning* (pp. 275–283). San Francisco, USA: Morgan Kaufmann.

Kotsiantis, S. (2011). Combining bagging, boosting, rotation forest and random subspace methods. *Artificial Intelligence Review, 35*, 223-240.

Kuhn, M., Wing, J., Weston, S., Williams, A., Keefer, C., Engelhardt, A., Cooper, T., Mayer, Z., Kenkel, B., Benesty, M., Lescarbeau, R., Ziem, A., Scrucca, L., Tang, Y., Candan, C., & Hunt, T. (2016). *Package 'caret'*. Retrieved from ftp://cran.r-project.org/pub/R/web/packages/caret/caret.pdf

Kuncheva, L. I., & Whitaker, C. J. (2003). Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning 51*(2), 181–207.

Leblanc, M., & Tibshirani, R. (1996). Combining estimates in regression and classification. *J. Am. Stat. Assoc. 91*(436), 1641–1650.

Le Roux, B. and Rouanet, H. (2004). *Geometric Data Analysis, From Correspondence Analysis to Structured Data Analysis.* Dordrecht. Kluwer.

Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Löfström, T., Johansson, U. & Bostrom, H. (2008). On the use of accuracy and diversity measures for evaluating and selecting ensembles of classifiers. *In Proceedings of the 7th International Conference on Machine Learning and Applications, San Diego, CA, USA*, 127–132.

Machova, K., & Barcak, F. (2006). A bagging method using decision trees in the role of base classifiers. *Acta Polytechnica Hungarica, 3*(2), 121–132.

Maclin, R. (1997). An empirical evaluation of bagging and boosting. *In Proceedings of the 14th National Conference on Artificial Intelligence, Providence, Rhode Island*, 546–551.

Marée, R., Geurts, P., Piater, J., & Wehenkel, L. (2004). A generic approach for image classification based on decision tree ensembles and local sub-windows. *Proceedings of the 6th Asian Conference on Computer Vision, 2,* 860–865.

Parvin, H., & Alizadeh, H. (2011). Classifier ensemble based class weighting. *American Journal of Scientific Research, 19*, 84–90.

Pearson, K. (1901). On Lines and Planes of Closest Fit to Systems of Points in Space. *Philosophical Magazine, 2* (11), 559–572.

R Core Team. (2016). R: A language and environment for statistical computing. *R Foundation for Statistical Computing, Vienna, Austria*. https://www.R-project.org/.

Rodríguez, J.J., Kuncheva, L.I., & Alonso, C.J. (2006). Rotation forest: a new classifier ensemble method. *IEEE Trans. Pattern Anal. Mach. Intell. 28*(10), 1619–1630.

Rodríguez, J. J., & Maudes, J. (2008). Boosting recombined weak classifiers. *Pattern Recognition Letters, 29*(8), 1049–1059.

Ron, K. (1996). Scaling Up the Accuracy of Naive-Bayes Classifiers: a Decision-Tree Hybrid. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*.

Ruta, D., & Gabrys, B. (2005). Classifier selection for majority voting. *Information Fusion 6*(1), 63–81.

Seewald, A. K. (2002). How to make stacking better and faster while also taking care of an unknown weakness. *In Proceedings of the 19th International Conference on Machine Learning, Sydney, Australia,* 554–561.

Simm, J., & Magrans de Abril, I. (2015). *Package 'extraTrees'*. Retrieved from https://cran.r-project.org/web/packages/extraTrees/extraTrees.pdf

Simon, D., (1992) Package 'glm'. Retrieved from https://stat.ethz.ch/R-manual/R-devel/library/stats/html/glm.html

Schapire, R.E., Freund, Y., & Bartlett, P., & Lee, W. S. (1998). Boosting the margin: a new explanation for the effectiveness of voting methods. *The Annuals of Statistics, 26*, 1651–1686.

Schapire, R.E., & Singer, Y. (1999). Improved boosting algorithms using confidence-rated predictions. *Machine Learning, 37*, 297–336

Skurichina, M., & Duin, R. P. (1998). Bagging for linear classifiers. *Pattern Recognition 31*(7), 909–930.

Skurichina, M., Kuncheva, L. I., & Duin, R. P. (2002). Bagging and boosting for the nearest mean classifier: effects of sample size on diversity and accuracy. *In Proceedings of the Third International Workshop on Multiple Classifier Systems, Cagliari, Italy,* 62–71.

Song, J., Wang, H. (2016). *Package 'Ckmeans.1d.dp'*. Retrieved from https://cran.r-project.org/web/packages/Ckmeans.1d.dp/Ckmeans.1d.dp.pdf

Thuraisingham, R., Tran, Y., Boord, P., and Craig, A. (2007). Analysis of Eyes Open, Eye Closed EEG Signals Using Second-Order Difference Plot. *Medical and Biological Engineering and Computing, 45*(12).

Ting, K., & Witten, I. (1999). Issues in stacked generalization. *Artificial Intelligence Research, 10*, 271–289.

Tsoumakas, G., Partalas, I., & Vlahavas, I. (2008). A taxonomy and short review of ensemble selection. *ECAI: Workshop on Supervised and Unsupervised Ensemble Methods and their Applications (SUEMA-2008),* 41-46.

Urbanek, S. (2016). Package 'rJava'. Retrieved from https://cran.r-project.org/web/packages/rJava/rJava.pdf

Valentini, G. (2004). Random aggregated and bagged ensembles of SVMs: an empirical bias-variance analysis. *International Workshop Multiple Classifier Systems, Lecture Notes in Computer Science 3077,* 263–272.

Webb, G., & Conilione, P. (2003). *Estimating bias and variance from data.* Technical report, School of Computer Science and Software Engineering, Monash University.

Wehenkel, L. (1997). Discretization of continuous attributes for supervised learning: variance evaluation and variance reduction. *Proceedings of the International Fuzzy Systems Association World Congress,* 381–388.

Wolpert, D. (1992). Stacked Generalization. *Neural Networks, 5*(2), 241-259.

Yeh, C., & Lien, H. (2009). The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications, 36*(2), 2473-2480.

Zeng, X., Chao, S., & Wong, F. (2010). Optimization of bagging classifiers based on SBCB algorithm. *In Proceedings of the International Conference on Machine Learning and Cybernetics, Qingdao, China,* 262–267.

Zhang, P., Zhu, X., Shi, Y., Guo, L., & Wu, X. (2011). Robust ensemble learning for mining noisy data streams. *Decision Support Systems, 50*(2), 469-479.

Zhang, C., & Zhang, J. (2009). A novel method for constructing ensemble classifiers.

Statistics and Computing, 19 (3), 317-327.

Zheng Z., and Padmanabhan B. (2007). Constructing Ensembles from Data Envelopment Analysis. *INFORMS Journal on Computing,* 19(4), 486–496.

Zhu, H., Beling, P., & Overstreet, G. (2002). A Bayesian framework for the combination of classifier outputs. *The Journal of the Operational Research Society, 53*(7), 719–727.

Zhu, D. (2010). A hybrid approach for efficient ensembles. *Decision Support Systems, 48*(3), 480-487.